BENSON REVISITED: THE CASE AGAINST PATENT PROTECTION FOR ALGORITHMS AND OTHER COMPUTER PROGRAM-RELATED INVENTIONS

*Pamela Samuelson**

Abstract

For the purpose of clarifying the public policy debate over the proper role for patents in the protection of computer program innovations, this article revisits the 1972 Supreme Court decision in *Gottschalk v. Benson.*[1] It argues that although the *Benson* Court did not clearly articulate the rationale for its decision, there is a basis in patent law for denying patents to computer program algorithms and to a number of other computer program-related innovations.[2] The author takes issue with Professor

---

* Professor of Law, University of Pittsburgh School of Law. The author wishes to thank the following people for their insightful comments on earlier drafts of this paper: Professors Martin J. Adelman, Ralph Brown, Peter Jaczi, Dennis Karjala, Robert Merges, and Jerome Reichman; attorneys James Dabney, Brian Kahin, Steven Lundberg, and John Sumner; and technologists John Hammer, Hans Oser, and Richard Stallman. Most of all she thanks her husband Robert J. Glushko for his technical expertise, his many editorial suggestions, and his patience during the time this article was being written.

[1] 409 U.S. 63 (1972).

[2] Throughout this article, the author has chosen the term "computer program-related invention" or "program-related invention" to refer to the array of things for which patent applications might be filed. The term includes not only algorithms, but also other components of a program (such as a data structure, the modular design for a program, a design for accomplishing a particular function by program, user interface functionalities, and the like). It also includes claims for improved industrial processes that might include computer program components. It is the author's understanding that these are the sorts of program-related inventions which the U.S. Patent and Trademark Office [hereinafter Patent Office] considers under its current policy to be patentable subject matter when properly claimed.

While the term "program-related invention" is potentially broad enough to cover claims for such things as computer programming languages and even computer programs themselves, the author does not intend to include these things within this term, for she does not believe that the Patent Office currently regards either computer programs or programming languages to be patentable subject matter. The author is aware that Professor Chisum argues, based on *In re* Carver, 227 U.S.P.Q. (BNA) 465 (PTO Bd. App. 1985), that programs themselves should be patentable subject matter. *See* 1 D. CHISUM, CHISUM ON PATENTS § 1.02[4], at 1-28.2 (1990). However, current Patent Office policy does not accept claims tied to computer program source code instructions. *See Oversight Hearing on Computers and Intellectual Property Before the Subcomm. on Courts, Intellectual Property and the Administration of Justice of the House Judiciary Comm.*, 101st Cong., 2d Sess. 7-8 (Mar. 7, 1990) (statement of Jeffrey M. Samuels, Acting Commissioner of Patents and Trademarks) [hereinafter *Samuels Testimony*]; *see also PTO Report on Patentable Subject Matter: Mathematical Algorithms*

Chisum's conclusion that *Benson* should be overruled and questions the validity of distinctions, such as the one between "mathematical" and "nonmathematical" algorithms, drawn by courts in applying *Benson* to computer program innovations. The author also questions using the patent system to protect program innovations that lie in the representation, organization, manipulation, and display of information. That the computer software industry has grown significantly without patent protection and that many in the industry express opposition to or doubts about patent protection for software innovations suggest that we should be wary of a policy that would grant patents to any computer program-related innovation. Historical limitations on the scope of patents, both in the United States and abroad, and concerns raised by prominent people in the computer science and software development communities raise serious doubts about the use of patents as a form of intellectual property protection for software innovations.

TABLE OF CONTENTS

---

and Computer Programs, 38 Pat. Trademark & Copyright J. (BNA) 563, 569-71 (1989) [hereinafter *PTO Report*] (distinguishing between computer programs, which are not patentable, and computer processes, which are patentable).

I. INTRODUCTION

For most of the past twenty-five years, it was widely believed that computer program-related inventions were rarely, if ever, patentable.[3] The 1972 Supreme Court decision, *Gottschalk v. Benson*, in which the Court ruled that a computer program algorithm is not patentable,[4] contributed significantly to this view. This decision also seemed to call into question the patentability of other computer program innovations.[5] Two subsequent Supreme Court decisions, *Parker v. Flook*[6] in 1978 and *Diamond*

---

[3] *See, e.g.*, OFFICE OF TECHNOLOGY ASSESSMENT, COMPUTER SOFTWARE & INTELLECTUAL PROPERTY: BACKGROUND PAPER 8 (1990) [hereinafter OTA BACKGROUND PAPER]; *see also* NATIONAL COMMISSION ON NEW TECHNOLOGICAL USES OF COPYRIGHTED WORKS, FINAL REPORT 16-17 (1979) [hereinafter CONTU FINAL REPORT] (stating that it is still unclear whether a patent could ever be obtained for a computer program). This Commission [hereinafter CONTU] recommended copyright as a form of intellectual property protection for computer programs in part because of its perception that patents would rarely, if ever, be available for program innovations. *Id.*; *see also* REPORT OF THE PRESIDENT'S COMMISSION ON THE PATENT SYSTEM, "TO PROMOTE THE PROGRESS OF . . . USEFUL ARTS" IN AN AGE OF EXPLODING TECHNOLOGY 13 (1966) [hereinafter 1966 REPORT] (recommending against patent protection for computer programs in part because of the apparent availability of copyright protection for computer programs).

Although the debate on software patents so far has focused primarily on "subject matter" problems with patenting program-related inventions, it has long been apparent that the novelty and nonobviousness requirements for patenting these innovations also would tend to limit significantly the availability of patents for programs. *See, e.g.*, Davidson, *Protecting Computer Software: A Comprehensive Analysis*, 23 JURIMETRICS J. 339, 357 (1983).

One other patent system mechanism protecting the public from excessive numbers of patents being issued is the claim examination process, in which limitations on the scope of claims may be added as a patent application is prosecuted in the Patent Office. As the patent examiner explores the prior art pertinent to a particular application, he or she may insist that the applicant add limitations to the claims so that the patent claims only cover the real contribution to the state of the art made by the patent applicant. In general, the more words that are added to patent claims, the less broad is the scope of the claims. Overbroad claims that are mistakenly allowed by patent examiners can be struck as invalid because of their overbreadth. *See infra* notes 100-01 and accompanying text. Many in the software industry are concerned about the ability of the Patent Office to make appropriate judgments about novelty and nonobviousness and to limit claims to an appropriate scope. *See infra* notes 441-45 and accompanying text.

[4] 409 U.S. 63 (1972). The legal issue before the Court in *Benson*, and in most of the subsequent cases on the patentability of computer program-related inventions, was whether the claimed invention (in *Benson*, an algorithm for converting binary coded decimals to pure binary form) was a "process" that was patentable under the patent statute. *See* 35 U.S.C. § 101 (1988); *see also infra* notes 102-05 and accompanying text.

[5] *See infra* notes 111-14 and accompanying text.

[6] 437 U.S. 584 (1978); *see infra* notes 201-13 and accompanying text.

v. Diehr[7] in 1981, reaffirmed the *Benson* ruling on the unpatentability of algorithms.[8] Even though the Supreme Court did conclude that Diehr's invention was patentable, the *Diehr* decision was regarded as a very limited one for many years. It is limited in that it affirms only that patents can issue for traditionally patentable industrial processes which include a computer program as an element.[9]

Despite the consistency in the Supreme Court's rulings against the patenting of algorithms, the Patent Office is now issuing patents for a wide variety of non-industrial computer program-related inventions and even seems to be issuing patents for computer program algorithms.[10] Some patent lawyers argue that this change is consistent with *Diehr*, which they read to hold that only claims for "unapplied" algorithms are unpatentable.[11] Professor Donald Chisum, the prominent patent scholar, attacks the Supreme Court case law more directly by calling for *Benson* to be overruled and for patent law to embrace the patentability of algorithms and other computer program-related inventions.[12]

The purpose of this article is to restate the case against patent protection for algorithms and many other computer program-related inven-

---

[7] 450 U.S. 175 (1981); *see infra* notes 260-81 and accompanying text.

[8] Reaffirmation of the *Benson* holding can be found in *Flook*, at 437 U.S. at 588-89, and in *Diehr*, at 450 U.S. at 185-86.

[9] The majority characterized the patentable process in *Diehr* as a method for curing synthetic rubber. *Diehr*, 450 U.S. at 177. A computer program was a component in the claimed process, but this program was not claimed as *the* patentable process. For a discussion of the various ways in which members of the Court characterized Diehr's process and the different implications they drew from their varying characterizations of the nature of the invention in that case, see *infra* notes 260-63 and accompanying text. For information concerning the narrow interpretation long given to *Diehr*, see, for example, OFFICE OF TECHNOLOGY ASSESSMENT, INTELLECTUAL PROPERTY RIGHTS IN AN AGE OF ELECTRONICS AND INFORMATION 85-87 (1986) [hereinafter OTA REPORT].

[10] The Patent Office recently issued a report aimed at giving guidance on the patentability of computer program-related inventions. *See PTO Report*, *supra* note 2. This report states that mathematical algorithms "per se" are not patentable. *Id.* at 564. However, for reference to and discussion of examples of some algorithm patents that have issued in recent years, see *infra* notes 282-94 and accompanying text.

[11] *See, e.g.*, Smith, Yoches, & Anzalone, *Computer Program Patents*, COMPUTER LAW., Apr. 1988, at 1, 3 (arguing that one must simply be careful in drafting the claims to obtain a patent on an algorithm); Sumner & Lundberg, *The Versatility of Software Patent Protection: From Subroutines to Look and Feel*, COMPUTER LAW., June 1986, at 1, 3 (arguing that *Diehr* makes only unapplied algorithms unpatentable).

[12] *See* Chisum, *The Patentability of Algorithms*, 47 U. PITT. L. REV. 959, 971 (1986). Professor Chisum's thesis is discussed and criticized *infra* notes 296-345 and accompanying text.

tions[13] in order to clarify the legal and public policy debates on this important subject.[14] The author finds a substantial basis in patent law for *Benson*'s ruling that computer program algorithms are unpatentable and for rejection of patents for many other program-related innovations. Both the appellate court case law on program patentability and the recently issued Patent Office guidelines lack a sound theoretical basis and rely heavily on indefensible distinctions. This article explains the basis for these assertions and analyzes recent appellate court decisions that have created a contradictory line of decisions on program-related patentability. These decisions have brought the debate about patentability of program-related inventions around full circle to where it began more than twenty years ago: The appellate court that reviews the Patent Office's decisions has come to adopt the Patent Office's first rationale for rejecting claims for program-related inventions, but has not acknowledged that it has done so.

In order to have a clear view of where the law now stands on the patentability of computer program innovations, therefore, it is necessary to go back to its historical roots. Part II of this article reviews the terms of the debate over computer program patentability in the years before the Supreme Court decision in *Benson*. Part III discusses the *Benson* case and how and why the Supreme Court's decision changed the nature of the debate over the patentability of program innovations. Part IV reviews the myriad ways in which the Court of Customs and Patent Appeals (CCPA) evolved in its interpretation of *Benson* in the years before the Supreme Court's *Diehr* decision. Part V analyzes the *Diehr* decision, asks whether

---

[13] The case against patents for computer program innovations has been made before. *See infra* notes 40-52 and accompanying text; *see also Oversight Hearing on Computers and Intellectual Property Before the Subcomm. on Courts, Intellectual Property and the Administration of Justic of the House Judiciary Comm.*, 101st Cong., 1st Sess. (Nov. 8, 1989) (statement of Leo J. Raskind).

[14] Elsewhere the author has argued that if both patent and copyright protection are to be available for computer program innovations, it is important to define the boundaries of and relationships between these two laws as applied to computer programs. *See, e.g., Oversight Hearing on Computers and Intellectual Property Before the Subcomm. on Courts, Intellectual Property and the Administration of Justice of the House Judiciary Comm.*, 101st Cong., 1st Sess. (Nov. 8, 1989) [hereinafter *Samuelson Testimony*]; Samuelson, *Is Copyright Law Steering the Right Course?*, IEEE SOFTWARE, Sept. 1988, at 78 [hereinafter Samuelson, *Right Course*]; Samuelson, *Survey on the Patent/Copyright Interface for Computer Programs*, 17 AIPLA Q.J. 256 (1989) [hereinafter *Samuelson Survey*]; Samuelson, *Why The Look and Feel of Software User Interfaces Should Not Be Protected by Copyright Law*, 32 COMMUN. ACM 563 (1989) [hereinafter Samuelson, *Look and Feel*]. However, this merely begs the question whether patents are in fact an appropriate form of intellectual property protection for programs.

some of the recently issued program algorithm patents are consistent with *Diehr*, and analyzes Professor Chisum's argument in favor of granting patents to algorithms and overruling *Benson*. Part VI discusses recent case law that raises serious doubts about the patentability of data-processing innovations. This part also establishes the link between patentability questions raised by claims for "mathematical algorithms" and claims for algorithms for processing other kinds of data. The unusual nature of computer programs is to blame for the complex problems that computer program innovations present for the patent system.

Parts II through VI contain very little public policy analysis of the issues raised by computer program patents, in part because the primary aim of these sections is to elucidate the case law on program patentability, and in part because the case law itself is remarkably empty of such analysis. Since the patentability of computer program innovations is a matter of considerable public policy importance, Part VII explores the current public policy debate over such patents. There is substantial opposition to patenting program innovations from within the computer science community[15] and the software industry.[16] If the software industry neither wants

---

[15] *See, e.g.*, Newell, *Response: The Models Are Broken! The Models Are Broken!*, 47 U. PITT. L. REV. 1023 (1986). Professor Newell's article was prepared in response to Professor Chisum's article. *See* Chisum, *supra* note 12. Professor Newell is the author of numerous books, among them: S. CARD, T. MORAN & A. NEWELL, THE PSYCHOLOGY OF HUMAN-COMPUTER INTERACTION (1983); A. NEWELL, COMPUTER STRUCTURES: PRINCIPLES AND EXAMPLES (1982); A. NEWELL & H. SIMON, HUMAN PROBLEM-SOLVING (1972).

[16] *See, e.g.*, Doler, *Experts Fear That Companies Misuse Patent, Copyright Protection Laws*, PC WEEK, May 1, 1989, at 67; Gibbons, *Patents Throw Obstacles in the Way of Progress*, PC WEEK, Sept. 18, 1989, at 77; *Software Patents: Law of the Jungle*, ECONOMIST, Aug. 18, 1990, at 59; Burton, *Can U.S. Software Industry Hold its Lead?*, Investor's Daily, Apr. 17, 1990, at 1, col. 4; *Software Industry in Uproar Over Recent Rush of Patents*, N.Y. Times, May 12, 1989, at A1, col. 5; *Will Software Patents Cramp Creativity?*, Wall St. J., Mar. 14, 1989, at B1, col. 3. On March 23, 1989, the Massachusetts Institute of Technology sponsored a four-hour program entitled "Software Patents: A Horrible Mistake?" Several speakers at the program, including Dan Bricklin, the developer of the first electronic spreadsheet program VisiCalc, and Duff Thompson, the general counsel of WordPerfect, answered the question posed by the program title with a resounding "yes." (Also on the program was the general counsel of a major computer company, speaking in favor of patents for computer program innovations.) *See* Seminar Notes, MIT Communications Forum, *Software Patents: A Horrible Mistake?*, Mar. 23, 1989 [hereinafter MIT Notes]; *see also* Samuelson & Glushko, *Comparing the Views of Lawyers and User Interface Designers on the Software Copyright "Look and Feel" Lawsuits*, 30 JURIMETRICS J. 121, 140 (1989) (survey showing opposition to patent protection for various aspects of computer programs, including algorithms).

Although patent lawyers seem to favor patent protection for computer program innovations strongly, *see, e.g.*, sources cited *supra* note 11, intellectual property law is supposed to give incentives

nor needs the patent system in order to be a vital and innovative industry, then, as a matter of public policy, it is sensible not to use the patent system for the protection of program-related innovations. A clear-headed review of the patent case law suggests that there is ample basis in patent law for withholding such protection.

II. THE BATTLE BETWEEN THE PATENT OFFICE AND THE CCPA BEFORE *GOTTSCHALK V. BENSON*

In order for an invention to be patented, it must meet certain standards.[17] One of these standards is that the invention be for one of the kinds of "subject matter" which Congress designated as eligible for patenting.[18] The patent statute names four categories of patentable "subject matter": machines, manufactures, compositions of matter, and processes.[19] Once one gets over the "subject matter" hurdle, there are still a number of other requirements to be satisfied. These requirements include applying for the issuance of a patent[20] and demonstrating to a patent examiner that the claimed invention is both "novel"[21] (not found in the state of the art)

---

to invest in innovation to those who develop computer software — not to patent lawyers. Thus, it is appropriate to give serious consideration to the concerns being voiced within the industry about software-related patents, rather than to acquiesce passively in the apparent *fait accompli* of a transformed Patent Office policy.

[17] In the Anglo-American tradition, the right to assert an entitlement to exclude others from utilizing one's innovation is not considered to be a natural right that arises as a result of the creative act (although if it can successfully be kept secret, it can generally be protected as a trade secret). Rather, this right arises as a matter of statutory (or positive) grant, and then only if the claimant follows certain procedures, her work meets certain standards, and, in the case of patents, she makes certain disclosures about the invention. *See* OTA REPORT, *supra* note 9, at 38; *see also* Gordon, *An Inquiry Into the Merits of Copyright: The Challenges of Consistency, Consent, and Encouragement Theory*, 41 STAN. L. REV. 1343 (1989) (discussing natural rights theory).

[18] For discussion of the desirability of a more open-ended subject matter standard for patent law, see Anderson, *Statutory Subject Matter in Intellectual Property: Application of the Copyright Scheme to the Patent System*, 20 S. TEX. L.J. 161 (1979). The author discusses some of the dangers of having a subject matter provision for patent law that is too open-ended *infra* notes 398-426 and accompanying text.

[19] 35 U.S.C. § 101 (1988); *see generally* 1 D. CHISUM, *supra* note 2, §§ 1.01-.06 (discussing the subject matter eligible for patent protection).

[20] 35 U.S.C. § 111 (1988). *See, e.g.*, 2 D. CHISUM, *supra* note 2, §§ 7.01-8.06 (discussing the disclosure and claim-drafting requirements of patent law); 3 D. CHISUM, *supra* note 2, §§ 11.01-.07 (discussing Patent Office procedures for obtaining a patent).

[21] 35 U.S.C. § 102 (1988). Although the primary meaning of "novel" relates to whether something is already in the state of the art as of a particular time, the term has a set of specific and

and "inventive"[22] (a nonobvious advance over the state of the prior art).

The debate over patentability of algorithms (and of other computer program innovations) is really a debate over whether an algorithm is the sort of "process" that Congress meant to include within the reach of the patent statute. This debate, in turn, needs to be understood in the context of a long history of interpreting the meaning of the term "process."[23] Algorithms are neither the first nor the only subject matter over which questions have been raised about whether it is a patentable "process."

A. *Interpreting "Process" More Narrowly Than Its Ordinary Meaning For Patent Purposes*

Upon reflection, it is obvious that not everything which is a "process," in the ordinary sense of the word, is meant to be a patentable kind of process. For example, the processes of reading a book, of interpreting a book, and of writing a book are unpatentable processes. The reason these processes are not patentable is not that it is impossible for a person to do these things inventively, but that Congress meant for the word "process," as it is used in the patent statute, to have a narrower meaning than the ordinary usage of the word might suggest.[24]

---

technically precise meanings that section 102 details. *See, e.g.*, 1 D. CHISUM, *supra* note 2, §§ 3.01-.08 (discussing the novelty requirement of patent law).

[22] 35 U.S.C. § 103 (1988); *see, e.g.*, 2 D. CHISUM, *supra* note 2, §§ 5.01-.06 (discussing the "nonobviousness" standard of patentability). For an analysis of recent case law arguing that the case law standard for what is "nonobvious" enough to deserve patent protection is not consistent with legislative intent or the fulfillment of the economic objectives of patent law, see Merges, *Commercial Success and Patent Standards: Economic Perspectives on Innovation*, 76 CALIF. L. REV. 803 (1988).

[23] Some of the debates over the meaning of "process" include:

(1) whether a process which is the inherent function of a machine is patentable, 1 D. CHISUM, *supra* note 2, § 1.03[2];

(2) whether medical procedures are patentable processes, *id.* § 1.03[3];

(3) whether agricultural methods are patentable processes, *id.*;

(4) whether methods requiring some human reaction (aesthetic, emotional, or normative) could be patentable, *id.* § 1.03[4];

(5) whether business methods are patentable, *id.* § 1.03[5]; and

(6) whether mental processes or processes involving mental steps could be patented, *id.* § 1.03[6].

[24] One major constraint on the meaning of the term "process" in the patent statute can be traced to the U.S. Constitution and its provision empowering the Congress to enact legislation to promote the "useful arts" by granting exclusive rights to inventors for limited times. "Useful arts" is understood to be the realm of technological and industrial improvements. For further discussion of this limitation,

How much narrower Congress intended the meaning of "process" to be is the critical issue, and it is one over which many words had been written prior to the battle between the Patent Office and the CCPA over the patentability of algorithms and other software innovations. The patent rules against the patenting of "business methods"[25] and of processes calling for subjective judgments[26] are two manifestations of this narrow construction of the term "process." However, in determining the patentability of computer programs, the chief doctrine of concern is the "mental process" or "mental steps" rule.

B. The "Mental Steps" Doctrine

In a series of decisions rendered over several decades before computer programs surfaced as a problem area, the Patent Office Board of Appeals and the CCPA interpreted the term "process" as not intended to cover "mental processes: processes consisting in whole or substantial part of 'mental steps.' "[27] These decisions involved claims for patenting processes

---

see *infra* notes 343-44 and accompanying text.

[25] For a discussion of the older cases, see, for example, 1 D. CHISUM, *supra* note 2, § 1.03[5]. One rationale for the rule against patenting business methods is that improved business methods are not contributions to the "useful arts," as that term has been understood. *See, e.g.*, Note, *The Patentability of Computer Programs: Merrill Lynch's Patent for a Financial Services System*, 59 IND. L.J. 633 (1984).

Another rationale may be that there has not been a need for patent incentives to bring about an appropriate level of innovation in business methods. Michael Milken, for example, seems to have had ample economic incentives and reward for introducing "junk bond" offerings to Wall Street without also getting a patent for this innovation.

[26] *See, e.g.*, 1 D. CHISUM, *supra* note 2, § 1.03[4] (discussing a case in which the claimant tried to patent an advertising scheme). Inventive ideas that call for subjective reactions or judgments should be distinguished from the "mental processes" or processes with "mental steps" which are discussed *infra* notes 27-38 and accompanying text. Subjective processes are unpatentable because they often do not relate to the "technological arts," and, even if they do, they are not deterministic (*i.e.*, following the method will not necessarily lead to the same results because of different subjective reactions or judgments made by different users) in the way that much of what has been sought to be patented under the "mental steps" doctrine is deterministic. Therefore, it is difficult to draft claims for them.

[27] *See generally* 1 D. CHISUM, *supra* note 2, § 1.06[6]. The earliest of the "mental steps" cases, according to Chisum, is *Ex parte* Meinhardt, 1907 Comm'n Dec. 237 (cited in 1 D. CHISUM, *supra* note 2, § 1.03[6], at 1-78.1). The main cases involving the "mental steps" doctrine are: *In re* Abrams, 188 F.2d 165 (C.C.P.A. 1951) (finding method of prospecting for oil deposits not patentable because the heart of the invention was in its mental steps); *In re* Shao Wen Yuan, 188 F.2d 377 (C.C.P.A. 1951) (finding mathematical means to determine optimal profile of airfoil exhibiting desired aerodynamic characteristics not patentable); *In re* Heritage, 150 F.2d 554 (C.C.P.A. 1945) (finding method

in which human beings took measurements about something, and after making calculations with data derived from these measurements, learned useful information about how to solve a problem in a technological field.[28] The measurements, calculations, and interpretations of data are the "mental processes" or "mental steps" to which the cases refer.[29]

One of these cases, *In re Abrams*, endorsed a set of "rules" by which to judge processes involving mental steps.[30] The first rule is that if a process is "purely mental," it is not patentable.[31] The second is that if a process contains both mental and physical steps, but the advance over the prior art is found in the mental steps, it too is not patentable.[32] The third is that if

---

of color-coating fiber board not patentable because the only novel feature of the method was in its process of selecting the amount of coating material to be used); Halliburton Oil Well Cementing Co. v. Walker, 146 F.2d 817 (9th Cir. 1944) (finding method of computing to determine depth of oil well not patentable), *rev'd on other grounds*, 329 U.S. 1 (1946); Don Lee, Inc. v. Walker, 61 F.2d 58 (9th Cir. 1932) (finding formula for computing the centrifugal force of engine shafts to determine the appropriate mass and positions of counterbalances not patentable).

These, however, are not the only "mental step" or "mental process" cases. *See* Diamond v. Diehr, 450 U.S. 175, 195-96 nn. 6-9 (1981) (Stevens, J., dissenting) (citing other "mental step" or mental process" cases); 1 D. CHISUM, *supra* note 2, § 1.03[6]. In *Benson*, the Supreme Court endorsed the view that mental processes are not patentable. *See infra* note 98; *see also infra* notes 42-47, 347-76, and accompanying texts (discussing mental process issues raised by computer program patents). Possible rationales for the mental process and mental steps rules are discussed *infra* note 34.

[28] *See, e.g., In re* Abrams, 188 F.2d 165 (C.C.P.A. 1951).

[29] *See, e.g., id.* at 167 (holding that method calling for "calculating," "comparing," "converting [data]," "determining," and "correcting" involved unpatentable mental steps); Halliburton Oil Well Cementing Co. v. Walker, 146 F.2d 817, 821 (9th Cir. 1944) (concluding that use of such descriptive words as "determining," "counting," "observing," "measuring," "comparing," and "computing" were fatal to the process claim), *rev'd on other grounds*, 329 U.S. 1 (1946).

[30] *Abrams*, 188 F.2d at 166. The "rules" were proposed by Abrams' counsel, who hoped to use them to persuade the CCPA that his client's application fell into the third category. This was the only category that Abrams' counsel thought was patentable. To the lawyer's chagrin, the CCPA ruled that Abrams' application fell into the second category. *Id.* at 170.

By making such direct and approving references to these rules, the CCPA seemed to give clear endorsement to them. This endorsement is noteworthy because later in the *Prater* and *Musgrave* decisions, the CCPA denied ever giving the rules any weight. *See infra* notes 64-65 and accompanying text. As the next three footnotes indicate, the Supreme Court's *Benson*, *Flook*, and *Diehr* decisions are consistent with the *Abrams* rules.

[31] *Abrams*, 188 F.2d at 166. This "rule" seems to cover Benson's claim for converting binary coded decimals to pure binary form, which the CCPA admitted could be done by hand. *See* Gottschalk v. Benson, 409 U.S. 63 (1972); *see also infra* note 86 and accompanying text.

[32] *Abrams*, 188 F.2d at 166. The *Flook* case involved a claim for a method of updating alarm limits in a catalytic conversion plant. Although the process contained at least one "physical" step (adjusting the plant equipment to reflect the updated alarm limit), the only new aspect of the process was in the equation which Flook had developed. *See* Parker v. Flook, 437 U.S. 584 (1978); *infra* note 195 and accompanying text.

both mental and physical steps have been claimed and there is some novelty in the physical as well as the mental steps, then the process is patentable.[33] The courts deciding *Abrams* and the other "mental process" cases, although they spoke frequently about "mental processes" and "mental steps," largely seemed to be concerned not with the "mental" character of the invention (all inventions, after all, are mental conceptions).[34] Instead,

---

A "point of novelty" test was endorsed not only in *Abrams* but also in earlier cases, such as *In re* Heritage, 150 F.2d 554 (C.C.P.A. 1945), and *In re* Shao Wen Yuan, 188 F.2d 377 (C.C.P.A. 1951) (decided about the same time as *Abrams*). The CCPA later acted as though the Patent Office had invented this test out of thin air and refused to acknowledge the roots of this doctrine in its own decisions. *See, e.g., In re* Chatfield, 545 F.2d 152, 158 (C.C.P.A. 1976).

[33] *Abrams*, 188 F.2d at 166. While the process in *Diehr* involved calculation steps to be performed by computer, the Court upheld the patentability of that process mainly because the process was said to yield an improved manufacture (perfectly cured rubber). *See* Diamond v. Diehr, 450 U.S. 175 (1981); *infra* notes 260-81 and accompanying text.

[34] The court in the *Abrams* case, however, was troubled about the mental character of the mental processes, saying: "Citation of authority in support of the principle that claims to mental concepts which constitute the very substance of an alleged invention are not patentable is unnecessary. It is self-evident that thought is not patentable." *Abrams*, 188 F.2d at 168. For a further discussion of the patentability of thoughts and other mental processes, see *infra* notes 391-94 and accompanying text.

It is unfortunate that this statement is virtually all that the court said to explain why mental processes should not be patentable. When the Supreme Court in *Benson* stated that mental processes are not patentable, the only explanation it gave is that mental processes are "basic tools of scientific and technological work." *Benson*, 409 U.S. at 67. Authors of two law review articles have argued that the "mental process" rule should be rejected. *See* McClaskey, *The Mental Process Doctrine: Its Origin, Legal Basis, and Scope*, 55 IOWA L. REV. 1148 (1970); Comment, *The Mental Steps Doctrine*, 48 TENN. L. REV. 903 (1981). McClaskey was a patent attorney for Bell Laboratories during the period when it was seeking patents for program-related inventions, including the algorithm contested in *Benson*. His main argument against the mental process rule is that persisting with this rule would retard progress in the field of computer programming. McClaskey, *supra*, at 1148-49. The Tennessee Comment is likewise concerned with limiting or eliminating this rule so that computer program innovations can be patented. Comment, *supra*, at 903, 917. These two articles take the lack of explanations as a sign that there is no sound reason for the doctrine. The author of this article believes that this lack of explanation instead may be a sign of how far outside the bounds of the patent system mental processes are perceived to be.

Apart from the "basic tools" argument the Supreme Court gave in *Benson*, one reason for the doctrine might be similar to the "business methods" rule discussed *supra* note 25, that mental processes are thought to be part of a set of "arts" different from the "technological arts." *See infra* notes 343-44 and accompanying text. Another might be a perceived lack of need for patent incentives to encourage people to invent new deterministic mental processes and to share them with others. A third might be the difficulty of enforcing patent rights in mental processes. It is one thing to grant a patent on a device such as a slide rule, for example, because there is a reasonable chance for a patentee to reap his or her rewards by controlling the sale of slide rules. In contrast, a process of converting measurements in inches to measurements in centimeters, however, is one without a product "bottleneck" to control, unless it be a chart, which then might run afoul of the "printed matter" rule,

these courts concentrated on not granting patent protection to data collection and analysis,[35] just as the courts deciding the "printed matter" cases concentrated on not granting patent protection to data representation or presentation.[36]

In the course of these decisions, so as to explain why these "mental processes" or processes with substantial "mental steps" are not patentable, the decisions sometimes make reference to some nineteenth-century patent cases, including an old Supreme Court case, *Cochrane v. Deener*, which defines "process" in the patent sense as including only those processes that transform matter from one physical state to another (as a chemical process

---

discussed *infra* note 36. Even controlling the sale of the charts, however, would not give control over all uses of the conversion process, for anyone who read the chart could use it undetected by the patentee. Perhaps another reason for the mental process doctrine is that since all humans think, the likelihood of independent invention of mental processes is greater than with technological areas in which there are fewer actors engaging in work to invent new ideas. For a further discussion of the patentability of thought processes, see *infra* notes 391-94 and accompanying text.

[35] Data analysis can certainly include numerical calculations, but can also involve a wide variety of other activities. As we will see, the CCPA has entirely ignored the larger data analysis issues presented in the computer program cases and concentrated only on issues involving numerical calculations. However, the CCPA found numerical calculation issues troublesome only after the *Benson* decision overturned its earlier, more receptive approach toward the patenting of equations.

[36] *See, e.g., In re* Rice, 132 F.2d 140 (C.C.P.A. 1942) (holding pictorial method of writing sheet music not patentable); *In re* Russell, 48 F.2d 668 (C.C.P.A. 1931) (holding method of arranging directories in a phonetic order not patentable); Guthrie v. Curlett, 10 F.2d 725 (2d Cir. 1926) (holding consolidated tariff index not patentable). In Boggs v. Robertson, 13 U.S.P.Q. (BNA) 214 (D.C. Cir. 1931), which involved a patent for a map projection system, the court regarded printed matter as unpatentable when it merely reduces an abstract idea to written form. *See generally* Note, *The Patentability of Printed Matter: Critique and Proposal*, 18 GEO. WASH. L. REV. 475 (1950).

As with the "business method" and "mental process" rules, there is little in the case law to explain the reasons for and scope of the "printed matter" rule. One reason for the "printed matter" rule may be a perception that although printing itself is a manufacturing process and part of the technological arts, the printed matter itself — and its contents, in particular — are not "in the technological arts," even when *about* the technological arts. A book describing how to organize one's work force in a rubber curing plant most effectively might be the product of a manufacturing process (*i.e.*, the book) and it might be about a manufacturing process, but the content of the work would still not be the kind of manufacture or process traditionally considered to be patentable.

Underlying the "printed matter" rule may be a perception that printed matter is among the set of things that are "writings" protectible by copyright law, not inventions in the "useful arts," and that copyright law strikes the appropriate balance between protection of expression and nonprotection of ideas for written texts. This balance would be disrupted if patents were available based on the content of the "printed matter." When "printed matter" has been patented, it has generally been in situations in which it has been integrated into some machine or physical structure which then supports the patent. *See, e.g.*, cases discussed in Note, *supra*, at 477-78. This Note suggests that the origins of the "printed matter" rule are interwoven with the "mental process" and "business methods" rules. *Id.* at 476.

does).[37] Some of these cases contained close questions as to whether the claim presented was for a mental process,[38] but the rule against patents for "mental processes" was quite well-established before the first computer program case arose.

C. *The Patent Office's Application of the "Mental Steps" Doctrine to Computer Programs*

In the mid-1960s, so many serious questions arose about the patent system's ability to deal with rapidly changing technologies that, in 1965, President Johnson appointed a special commission of distinguished scientists, academics, and representatives of leading computer and high-technology firms (as well as the Commissioner of Patents) to study the matter.[39] It was the Commission's judgment that patent protection for

---

[37] *Cochrane*, 94 U.S. 780 (1876) (cited in Halliburton Oil Well Cementing Co. v. Walker, 146 F.2d 817, 821 (9th Cir. 1944), *rev'd on other grounds*, 329 U.S. 1 (1946) and *In re* Abrams, 188 F.2d 165, 168 (C.C.P.A. 1951)). The most frequently cited part of *Cochrane*'s discussion of "process" is this description: "A process is a mode of treatment of certain materials to produce a given result. It is an act, or series of acts, performed upon the subject matter to be transformed and reduced to a different state or thing." 94 U.S. at 788.

Dr. Hanneman accepts the "transformation of matter" standard for what constitutes a patentable process under American law. *See* H. HANNEMAN, THE PATENTABILITY OF COMPUTER SOFTWARE 42 (1985); *see also* 1 W. ROBINSON, THE LAW OF PATENTS FOR USEFUL INVENTIONS § 166 (1890). For a discussion of the Supreme Court's use of *Cochrane v. Deener* in *Benson* and *Diehr*, see *infra* notes 102-05, 265-68, and accompanying texts.

[38] The "mental steps" or "mental process" doctrine is not without its line-drawing problems. *See, e.g.*, *Ex parte* McNabb, 127 U.S.P.Q. (BNA) 456 (PTO Bd. App. 1959). However, line-drawing is an inherent problem whenever one establishes a subject matter boundary for an intellectual property system. Its inherent purpose is to let the "right" things in and keep the "wrong" things out.

[39] The Commission was established by Executive Order No. 11,215, 30 Fed. Reg. 4661 (1965). Its membership was announced on July 23, 1965. It held 13 meetings, each of which lasted one to four days, for a total of 31 days of meetings. *See* 1966 REPORT, *supra* note 3, at ii. The members of the Commission were: John Bardeen, James W. Birkenstock, Howard W. Clement, Howard K. Nason, Sidney Neuman, Bernard Oliver, Harry Hunt Ransom, Simon Rifkind, Horton Guyford Stever, Charles B. Thornton, and the Commissioner of Patents, Edward J. Brenner, representing the U.S. Commerce Department; John M. Malloy, representing the Defense Department; Eugene J. Davidson, representing the Small Business Administration; and Charles F. Brown, representing the National Science Foundation. *Id.* at iv. John Bardeen was a recipient of a Nobel Prize for physics as the co-inventor of the transistor; James Birkenstock, at the time of his appointment to the Commission, was the vice president for commercial development for IBM Corporation; Howard Nason was president of Monsanto Research Corporation; Bernard Oliver was vice president for research and development at Hewlett-Packard Company; Harry Hunt Ransom was chancellor of the University of Texas; Horton Stever was president of Carnegie Mellon University; and Charles Thorton was chief

computer program innovations was not desirable. The Commission's 1966 report raised questions about the patentability of program inventions under the existing statute[40] and recommended legislation that expressly excluded computer programs from patent protection.[41]

Although the Patent Office had earlier seemed receptive to the idea of patenting program-related inventions that were properly claimed,[42] in 1966 it issued a set of guidelines that significantly limited the availability of patents for program-related inventions.[43] These guidelines provided

---

executive officer and chairman of the board of Litton Industries. The Commission made a number of suggestions for the improvement of the patent system in an age of "exploding technology."

[40] 1966 REPORT, *supra* note 3, at 12-13. According to the Report:

> Uncertainty now exists as to whether the statute permits a valid patent to be granted on programs. Direct attempts to patent programs have been rejected on the ground of nonstatutory subject matter. Indirect attempts to obtain patents and avoid the rejection, by drafting claims as a process, or a machine or components thereof programmed in a given manner, rather than as a program itself, have confused the issue further and should not be permitted.

Id. For a description of the reasons the Commission recommended against patent protection for programs, see *infra* notes 49-52 and accompanying text. *See also* H. HANNEMAN, *supra* note 37, at 120 (indicating that countries other than the United States used commissions to study the patentability of computer programs and that they reached much the same conclusion as the U.S. Commission). In none of its more than 30 computer program patentability decisions did the CCPA ever mention, let alone discuss or attempt to refute, this Commission's report.

[41] 1966 REPORT, *supra* note 3, at 12-13. In view of the fact that the Patent Office had issued guidelines that limited the instances in which patents would be available for computer program-related innovations, Congress probably felt it unnecessary to legislate against patents for program innovations.

[42] The initial approach taken by the Patent Office toward the patenting of program-related inventions did not focus on subject matter issues. In *Ex parte* King, 146 U.S.P.Q. (BNA) 590 (PTO Bd. App. 1964), for example, apparatus claims were made for a "digital processing machine comprising means for storing a string of digitally coded program syllables," which was to be performed "in accordance with a system of denoting mathematical expressions free of parenthetical groupings already known to mathematicians as the Polish Notation." *Id.* at 590-91. King's application was rejected on the ground that, as claimed, it "merely set forth the result or function accomplished by any computer operating on data in Polish Notation." *Id.* at 591. This was to be done, said the Board, "not through any novel structure set out in the claims but merely by the routine operation of storage and processing means normally found in stored program computers." *Id.* The Board's opinion suggests that, if claimed properly, and with truly novel subject matter, a programmed computer can be claimed as a new apparatus.

[43] 829 OFF. GAZ. PAT. OFFICE, Aug. 16, 1966, at 865. This was a proposed rule, which became effective in 1968. *See* 33 Fed. Reg. 15,609 (1968). None of the CCPA computer program decisions ever mentioned these guidelines. Despite the guidelines, some software patents did issue during this time period, such as a now-expired patent which IBM lawyers claim was for a spreadsheet program (claimed in apparatus form). *See* Patent No. 3,610,902, issued in 1971. As will become more clear later in this article, it has generally been easier to get a patent for a program-related invention if one

that regardless of whether they were claimed as apparatuses ("machines" in section 101 terms) or as methods ("processes" in section 101 terms), computer programs, as such, were not patentable subject matter.[44] The guidelines did not, however, make all computer program-related inventions unpatentable. Rather, they provided that a programmed computer could be claimed as a component of a patentable process if it was combined with nonobvious elements that produced a physical result.[45] The Patent Office relied on the "mental steps" doctrine and the *Cochrane* interpretation of "process"[46] to support these guidelines.[47] Its views also comported with the views of President Johnson's Commission.[48]

The Patent Office and the Presidential Commission gave several reasons in support of their policy recommendations. First, computer programs were thought not to be the kind of "processes" that Congress had intended to protect by passing the patent law.[49] Second, the art of computer programming, having emerged and flourished without reliance on patents, was believed not to need the kind of protection that patents would

---

claimed it in "apparatus" form. For examples of this technique, see *infra* notes 163, 378-86, and accompanying texts.

[44] As eventually adopted as official policy, the guidelines can be found at 33 Fed. Reg. at 15,610. Section 52 of the European Patent Convention codifies a rule against the patentability of computer programs as such. *See* H. HANNEMAN, *supra* note 37, at 242-47. International treatment of patent claims for computer program-related inventions is discussed further *infra* note 426. Even current Patent Office policy considers programs as such to be unpatentable. *See supra* note 2.

[45] The guidelines provide:

> [A] computer programming process which produces no more than a numerical, statistical or other informational result is not directed to patentable subject matter. Such a process may, however, form a part of a patentable invention if it is combined in an unobvious manner with physical steps of the character [such as] in the knitting of a pattern or the shaping of metal.

33 Fed. Reg. at 15,610. A fair reading of Diamond v. Diehr, 450 U.S. 175 (1981), *see infra* notes 260-81 and accompanying text, would suggest that the 1968 guidelines position is as far as the Supreme Court has been willing to go toward endorsing computer program patents. A similar rule also appears to predominate in most other countries that have considered the patentability of computer program innovations. *See infra* note 426.

[46] *See supra* notes 27-37 and accompanying text.

[47] *See* 33 Fed. Reg. at 15,610.

[48] 1966 REPORT, *supra* note 3. Since the Commissioner of Patents was a member of the Presidential Commission that drafted the 1966 Report, it is not surprising that there are similarities between the Patent Office's proposed guidelines and the Commission's Report. *See supra* note 39.

[49] For a discussion of the Patent Office's interpretation of the term "process" in the first set of computer program-related cases, see *infra* notes 60-61 and accompanying text. This was not an issue that the 1966 Report addressed.

offer.[50] Also, because both copyright and trade secret protection seemed to be available to protect programs, it appeared that patent protection was not needed.[51] In addition, the Patent Office had little of the relevant prior art in its files and little in-house expertise in programming to use for guidance in its assessment of patent claims directed to computer program implementations.[52] Thereafter, patent examiners increasingly began rejecting software patent applications on the ground that they failed to claim patentable subject matter.[53]

D. *The CCPA's Review of Computer Program-Related Inventions Prior to* Gottschalk v. Benson

Patent applicants whose computer program-related claims were rejected by the Patent Office found the CCPA to be a much more receptive audience to their pleas in favor of the patentability of their inventions.[54] In the

---

[50] *See* 1966 REPORT, *supra* note 3, at 13. The Patent Office's policy guidelines say nothing on this point.

[51] *Id.* For the Supreme Court's mention of this issue in *Benson*, see *infra* note 113. In 1964, the U.S. Copyright Office began accepting registration of computer programs as long as one deposited the full text of the source code with the Office. It did so, however, under its "rule of doubt," a rule which allows registration of materials as to which the Register of Copyrights has some doubt about copyrightability. The reason for doubting the copyrightability of computer programs was because of their mechanical character when in machine-readable form. *See* Samuelson, *CONTU Revisited: The Case Against Copyright Protection for Computer Programs in Machine Readable Form*, 1984 DUKE L.J. 663, 692-703 (discussing the evolution of copyright as a form of protection for computer programs).

The 1966 Report does not specifically mention trade secret protection for software, but it must nonetheless have been clear at the time that programs could be (and were) protected by trade secret law. *See* 1966 REPORT, *supra* note 3. The Patent Office policy statement makes no reference to the availability of other forms of protection in explaining its position on patents. *See* 33 Fed. Reg. 15,609-10 (1968); *see also PTO Report*, *supra* note 2.

[52] 1966 REPORT, *supra* note 3, at 13.

[53] In some cases, program-related claims were rejected for failure to satisfy section 112 requirements. *See, e.g., In re* Foster, 438 F.2d 1011 (C.C.P.A. 1971); *In re* Prater, 415 F.2d 1393 (C.C.P.A. 1969); *In re* Naquin, 398 F.2d 863 (C.C.P.A. 1968); *see also infra* note 67.

[54] If a patent examiner decides against the issuance of a patent, an applicant can appeal this decision, first, to the Patent Board of Appeals, and then to a Court of Appeals. 35 U.S.C. §§ 134, 141 (1988). Until October 1982, the CCPA heard appeals of Patent Office rejections. The CCPA, however, did not hear other kinds of patent appeals (*e.g.*, patent infringement appeals). These appeals were heard instead by the Circuit Courts of Appeals that heard most other federal cases. Since October 1982, all patent-related appeals have been heard by a specialized Court of Appeals, known as the Court of Appeals for the Federal Circuit. This court has only recently made any rulings about the patentability of computer program-related inventions. *See infra* notes 251-52 and accompanying text.

four years prior to the Supreme Court's decision in *Gottschalk v. Benson*,[55] there were eight reported cases on whether computer program-related inventions are patentable subject matter.[56] In all eight cases, the Patent Office rejected the claims as outside the subject matter bounds of the patent system, and the Patent Board of Appeals affirmed. In all eight cases, however, the CCPA reversed the Patent Office's rulings on subject matter grounds.

A curious thing about these eight pre-*Benson* cases is that none of them, not even the CCPA's decision in the *Benson* case, makes any more

---

[55] 409 U.S. 63 (1972).

[56] *In re* Waldbaum, 457 F.2d 997 (C.C.P.A. 1972) [hereinafter *Waldbaum I*] (holding method of determining the number of binary "1's" in electronic data patterns telephone system to be sufficiently technological in character to be patentable), *aff'd*, *In re* Waldbaum, 559 F.2d 611 (C.C.P.A. 1977) [hereinafter *Waldbaum II*] (rehearing *Waldbaum I* after *Benson* decision; in "affirming" *Waldbaum I*, the CCPA was just being cagey — it did not want to admit that it was really reversing the case); *In re* McIlroy, 442 F.2d 1397 (C.C.P.A. 1971) (upholding the patentability of a method of retrieving symbolic data from a stored string (usable, for example, in searching for words in a word-processor)); *In re* Benson, 441 F.2d 682 (C.C.P.A. 1971) (upholding the patentability of a method of converting binary coded decimals to pure binary form), *rev'd sub nom.* Gottschalk v. Benson, 409 U.S. 63 (1972); *In re* Foster, 438 F.2d 1011 (C.C.P.A. 1971) (upholding as patentable subject matter an improved method of processing seismographic data but affirming rejection of the claims as not limited to machine implementations); *In re* Musgrave, 431 F.2d 882 (C.C.P.A. 1970) (holding improved method of seismographic analysis for detecting subsurface formations to be patentable because it is "in the technological arts," even though it was apparently capable of being performed without the aid of a computer; *Musgrave*'s "technological arts" standard is discussed *infra* notes 66, 311-45, 393-405, and accompanying texts); *In re* Mahony, 421 F.2d 742 (C.C.P.A. 1970) (upholding the patentability of a method for "framing" electronic signals so that a computer can detect where each new character string begins; finding that it was not a mental process because of claim language referring to "bits" and "bit streams," construing this language to mean that the claims were limited to machine implementations); *In re* Bernhart, 417 F.2d 1395 (C.C.P.A. 1969) (upholding the patentability of apparatus and method claims for improved way to depict three-dimensional objects in two-dimensional form that used a set of equations to yield the proper coordinates because the claims covered more than just equations and the invention was to be carried out by machine); *In re* Prater, 415 F.2d 1378 (C.C.P.A. 1968) [hereinafter *Prater I*], *modified on rehearing*, 415 F.2d 1393 (C.C.P.A. 1969) [hereinafter *Prater II*] (affirming rejection of method claims for identifying the optimal set of equations with which to obtain accurate assessment of proportions of gaseous substances in spectographic analysis because of a failure to limit the claim language to machine implementations of the method).

Improved methods of seismographic analysis have been the most commonly litigated type of computer program-related invention. *See, e.g.*, *In re* Taner, 681 F.2d 787 (C.C.P.A. 1982); *In re* Walter, 618 F.2d 758 (C.C.P.A. 1980); *In re* Sherwood, 613 F.2d 809 (C.C.P.A. 1980), *cert. denied*, 450 U.S. 994 (1981); *In re* Johnson, 589 F.2d 1070 (C.C.P.A. 1978); *In re* Christensen, 478 F.2d 1392 (C.C.P.A. 1973), *overruled*, *In re* Taner, 681 F.2d 787 (C.C.P.A. 1982) (*Christensen* was not really overruled in *Taner*; it was officially reinterpreted in light of the *Diamond v. Diehr* decision; *see infra* note 198); *In re* Naquin, 398 F.2d 863 (C.C.P.A. 1968) (the only such case not to have been argued on subject matter grounds).

than an incidental use of the word "algorithm" in discussing the patentability issue.[57] Hence, none of the analysis contained in these lower court decisions focused on the patentability of "algorithms."[58] It was the Supreme Court's decision in *Benson* that shifted the focus of attention to "algorithms."[59]

While more will be said about *Benson* and the consequences of its focus on algorithms in subsequent parts of this article, the battle between the Patent Office and the CCPA was not about "algorithms." Rather, the main battle concerned the patentability of "mental processes." In this battle, the Patent Office maintained that if a process under evaluation could be performed by a person in his or her head or with the aid of pencil and

---

[57] *See, e.g., In re* Benson, 441 F.2d 682 (C.C.P.A. 1971) (referring incidentally to "algorithms" in descriptions of Patent Office actions), *rev'd sub nom.* Gottschalk v. Benson, 409 U.S. 63 (1972); *In re* Musgrave, 431 F.2d 882, 886 (C.C.P.A. 1970). Most of these eight decisions make no mention whatsoever of the term "algorithm."

[58] Although, prior to the Supreme Court's *Benson* decision, the Patent Office/CCPA debate over computer program-related inventions was not about "algorithms" (or even about equations, which is largely what the CCPA later interprets "algorithms in the *Benson* sense" to mean), four of the eight cases that the Patent Office found troublesome on subject matter grounds involved equations. *In re* Foster, 438 F.2d 1011 (C.C.P.A. 1971) (involving improved method of seismographic analysis for detecting subsurface formations); *In re* Musgrave, 431 F.2d 882 (C.C.P.A. 1970) (involving method of removing distortions from seismographic data to aid in detecting subsurface formations that used equations); *In re* Bernhart, 417 F.2d 1395 (C.C.P.A. 1969) (involving use of a set of equations in a method to convert data from three-dimensional into two-dimensional coordinates); *Prater I*, 415 F.2d 1378 (C.C.P.A. 1968) (involving method for identifying the optimal set of equations to use in spectrographic analysis to obtain accurate assessment of the proportions of gaseous substances), *modified on rehearing*, *Prater II*, 415 F.2d 1393 (C.C.P.A. 1969). The Benson claims were not expressed in the form of an equation, although they could have been.

In most of these cases, the only thing that distinguished the claimed invention from the prior art was the set of equations that the applicant had developed. *See, e.g., Bernhart*, 417 F.2d at 1401-02. In later cases, the "point of novelty" test for patentability of program-related inventions became a new point of contention between the CCPA and the Patent Office. *See infra* notes 189-215 and accompanying text.

The reason none of the claims in these cases was directed to equations alone was that a good patent lawyer in the late 1960s or early 1970s would have known that the Patent Office would be unreceptive to process claims drawn specifically to equations. Even the CCPA in *In re* Bernhart, 417 F.2d at 1399, rejected the idea of patenting equations. Because of this, a good patent lawyer would draft claims that did not appear to be for equations, or at least not solely for equations.

[59] After the Supreme Court's decision in *Benson*, the "mental process" rationale for rejecting program-related inventions rarely appears in the case law. *In re* Meyer, 688 F.2d 789 (C.C.P.A. 1982), and *In re* Sarkar, 588 F.2d 1330 (C.C.P.A. 1978), are two exceptions to this general proposition. In *Meyer*, the CCPA denied patent protection to a design for an expert system program because it mimed the mental steps that humans take in performing medical diagnoses. This case is discussed at length *infra* notes 363-73 and accompanying text.

paper, it was not a patentable process,[60] notwithstanding the fact that the patent applicant might only intend to assert rights to machine implementations of the process through use of a programmed digital computer. The Office was certain that such a process could not be patentable if not limited to machine implementations.[61]

There were two main phases of the CCPA's jurisprudence on the patenting of computer program-related innovations in the years before *Benson*, although the CCPA shifted its thinking numerous times in its quest for the "right" way to analyze claims for program-related inventions.[62] At

---

[60] One of the more interesting arguments made by the Patent Office in arguing for rehearing of the *Prater I* case, 415 F.2d 1378 (method of selecting equations for accurate spectrographic analysis), was its "first amendment" argument. The Office asserted that a patent on a mental process of this sort would "preclude people from thinking" or result in "mental infringement." The Office thought that this would interfere with rights protected by the first amendment. *Id.* at 1391-92 (Rich, J., dissenting from grant of rehearing). The CCPA did not directly address this issue in *Prater II*, although in a footnote, it quoted from an amicus brief by Bell Laboratories, which asserted, among other things, that "purely mental activity would not be [an] infringement since there would be no harm to the patentee, an essential element for any tort. In Bell's view, freedom of mind or thought would not necessarily be abridged by the grant of a patent with claims broad enough to [be] read on mental processes." *Prater II*, 415 F.2d at 1400 n.20.

The CCPA did, however, react to Patent Office assertions pertaining to artificial intelligence concepts in the *Bernhart* case. *Bernhart*, 417 F.2d 1395. The Patent Office's main argument against the patentability of Bernhart's claims for making two-dimensional depictions of three-dimensional objects was that the process could be done by a human without the aid of a machine. However, the CCPA said, in connection with the apparatus claim, that it knew "of no authority for holding that a human being, such as a draftsman, could ever be the equivalent of a machine disclosed in a patent application." *Id.* at 1399. As to the process claim, the court said, "To find that the claimed process could be done mentally would require us to hold that a human mind is a digital computer or its equivalent, and that a draftsman is a planar plotting apparatus or its equivalent. On the facts of this case we are unwilling so to hold." *Id.* at 1401.

That a human being is a kind of machine and that the human mind is (or is like) a digital computer are among the central theories of cognitive science and of the artificial intelligence disciplines. *See, e.g.*, J. HAUGELAND, ARTIFICIAL INTELLIGENCE: THE VERY IDEA (1985); *see also infra* notes 391-94 and accompanying text.

In *Bernhart*, the Patent Office also argued that an improved method for representing three-dimensional forms in two-dimensional space should be rejected for much the same reasons that "printed matter" has traditionally been rejected as nonstatutory subject matter. *Bernhart*, 417 F.2d at 1398. The CCPA, however, saw only a process that could be implemented by machine. This, it insisted, could be patented. *Id.* at 1399.

[61] *See, e.g., In re* Benson, 441 F.2d 682, 687-88 (C.C.P.A. 1971) (discussing the Patent Office's argument that Benson's process could be done with paper and pencil), *rev'd sub nom.* Gottschalk v. Benson, 409 U.S. 63 (1972).

[62] *See infra* note 70. One example of a short-lived shift in its thinking can be found in *Bernhart*, 417 F.2d 1395. The CCPA accepted the argument that "if a machine is programmed in a certain new and unobvious way, it is physically different from the machine without that program. . . . The fact

first, the CCPA upheld program-related claims as long as they contained some claim language that limited the patent scope to machine implementations of the process. Such language distinguished these claims from the old "mental process" cases of the past[63] that had not involved machine implementations.[64] Eventually, in *In re Musgrave*, the CCPA repudiated

---

that these physical changes are invisible to the eye should not tempt us to conclude that the machine has not been changed." *Id.* at 1400.

If each new computer program is perceived as a new machine, then each program would automatically be patentable subject matter. Curiously enough, this way of conceptualizing programs as new machines does not recur after *Bernhart*, even though it was among the clearest and most straightforward rationales for rejecting subject matter challenges to program-related inventions. This view of computer programs as creating new machines has some merit as a matter of computer science, but has been criticized as a focal point for patent analysis. *See, e.g.*, Gemignani, *Legal Protection for Computer Software: The View from '79*, 7 RUTGERS J. COMPUTERS, TECH., & L. 269, 278-81 (1980).

There are four other theories in favor of patentability for program-related inventions that were also not raised in the pre-*Benson* era:

(1) that programs, when executed in digital computers, satisfy the *Cochrane* standard because they cause electrons (a form of matter under at least some theories of the physical sciences) to be transformed;

(2) that programs satisfy the *Cochrane* standard because they cause computers to change from one state to another;

(3) that programs are patentable processes because they are new uses of old machines under 35 U.S.C. § 100(b) (1988); and

(4) that programs are patentable because an equivalent of any program can be constructed in hardware.

[63] For a discussion of the older "mental process" cases, see *supra* notes 27-38 and accompanying text. Three cases illustrate this first pre-*Benson* approach. *In re* Mahony, 421 F.2d 742 (C.C.P.A. 1970) (construing claim language to limit patent scope to machine implementations); *In re* Bernhart, 417 F.2d 1395 (C.C.P.A. 1969) (same); *Prater I*, 415 F.2d 1378 (C.C.P.A. 1968), *modified on rehearing*, *Prater II*, 415 F.2d 1393 (C.C.P.A. 1969) (affirming patent denial because of failure to limit claim language to machine implementations).

[64] Of the first phase cases before *Benson*, the *Prater* case contains the most extensive discussion of the "mental process" issue as it affects computer programs.

The CCPA in *Prater I* made a strong attack on the "mental process" objection to program-related inventions raised by the Patent Office. *Prater I*, 415 F.2d 1378. The CCPA asserted that its earlier decision, *In re* Abrams, 188 F.2d 165 (C.C.P.A. 1951), had been "misread," that the other "mental process" cases were "unsupported by precedent" and could have been decided on other grounds, and that the Supreme Court's decision in Cochrane v. Deener, 94 U.S. 780 (1876), had been "misconstrued." *Prater I*, 415 F.2d at 1386-88.

The CCPA stopped short of abolishing the "mental steps" limitation on patentability in *Prater I*, but purported to confine its application to situations in which the process could only be performed in the mind. *Id.* at 1389. Still, the court did question the viability of the doctrine. *Id.*

The Patent Office, arguing that complex policy issues were presented by the patentability of computer programs, petitioned for rehearing of the *Prater I* case. The CCPA was not impressed with the complex policy issues, for the court made no mention of them in *Prater II*, 415 F.2d 1393, and two judges explicitly denied that there were any such difficult or complex issues. *Prater II*, 415 F.2d at 1392-93. These two judges also denied that the CCPA was engaging in any policymaking in ruling on

the "mental process" doctrine altogether[65] and asserted that in order for a process to be patentable, it need only be part of the "technological arts."[66]

---

the program-related patent claims and asserted that the Patent Office had sought a rehearing solely because it disagreed with the CCPA's interpretation of the law. *Id.*

Nevertheless, the CCPA *was* making policy and not just applying existing law. They had to radically reconstrue the court's earlier decisions in order to find a basis for interpreting existing law to cover program-related inventions. During the entire debate over the patentability of program-related inventions, it was typical for the CCPA to act as if it were doing nothing out of the ordinary.

Over the objection of two of its members, the CCPA did decide to rehear the *Prater I* case, and upon rehearing, reversed its prior ruling. *Prater II*, 415 F.2d 1393. It did so, however, not because it found the subject matter of Prater's invention unpatentable, but because Prater had not limited his method claims to machine implementations. *Id.* at 1404. Although the CCPA repeated some of *Prater I*'s analysis of *Abrams* and the "mental steps" doctrine, the CCPA seemed to take a more moderate course in *Prater II*. It endorsed the view that "purely mental processes" and process claims not limited to machine implementations would be unpatentable. *Id.* at 1402-04. Not until *In re* Musgrave, 431 F.2d 882 (C.C.P.A. 1970), would this more moderate approach be dropped.

[65] *Musgrave* was the fourth CCPA decision on the patentability of computer program-related inventions. In *Musgrave*, the CCPA adopted a theme developed in *Prater I* and asserted that "the statutory language [of section 101] contains nothing whatever which would either include or exclude claims containing 'mental steps' and whatever law there may be on the subject cannot be attributed to Congress. It is purely a question of case law." *Musgrave*, 431 F.2d at 890. Finding the case law to be "something of a morass," the CCPA concluded that it could not "agree with the board that these claims . . . are directed to non-statutory processes merely because some or all the steps therein can also be carried out in or with the aid of the human mind or because it may be necessary for one performing the processes to think." *Id.* at 890, 893.

Musgrave, like *Prater I* before it, contains an extensive discussion of *In re* Abrams, 188 F.2d 165 (C.C.P.A. 1951), and a repudiation of its "so-called rules" for determining which "mental steps" cases might involve patentable inventions. *Musgrave*, 431 F.2d at 889-90. For a discussion of these rules, see *supra* notes 30-35 and accompanying text. *Musgrave* also discusses other case law on the patentability of "mental processes" and "mental steps." *Musgrave*, 431 F.2d at 891-92. *Musgrave* cites the *Prater* decisions as already having shown the error in the Patent Office's argument that *Cochrane* made processes performable by humans without the aid of a machine unpatentable. Id. at 893.

[66] The CCPA in *Musgrave* stated as follows: "All that is necessary, in our view, to make a sequence of operational steps a statutory 'process' within 35 U.S.C. § 101 is that it be in the technological arts so as to be in consonance with the Constitutional purpose to promote the progress of 'useful arts.'" *Musgrave*, 431 F.2d at 893. This patentability standard is discussed *infra* notes 311-45, 393-405, and accompanying texts.

The *Musgrave* decision, however, did not end the dispute between the Patent Office and the CCPA over patentability issues. In the four cases following *Musgrave* (but before the Supreme Court's *Benson* decision), the Patent Office continued to reassert its own definition of what was a patentable "process" and rejected computer program patent applications on the ground that they failed to claim statutory subject matter. *See Waldbaum I*, 457 F.2d 997 (C.C.P.A. 1972), *aff'd*, *Waldbaum II*, 559 F.2d 611 (C.C.P.A. 1977) (the CCPA did not really affirm *Waldbaum I*; *see supra* note 56); *In re* McIlroy, 442 F.2d 1397 (C.C.P.A. 1971); *In re* Benson, 441 F.2d 682 (C.C.P.A. 1971), *rev'd sub. nom.* Gottschalk v. Benson, 409 U.S. 63 (1972); *In re* Foster, 438 F.2d 1011 (C.C.P.A. 1971).

The CCPA opinions in these four cases reflect an increasing impatience with the Patent Office and its stubborn resistance to the CCPA's view of what constituted patentable subject matter. *See, e.g.*,

The CCPA was satisfied that computer program innovations met this requirement,[67] seemingly because they could be carried out by machine.[68]

Since the patent claims in these eight cases were generally not for programmed computer implementations (that is, were not claims for "machines") but for the processes themselves, the "mental process" issue seemed to be a serious one for the patent system. It called for a frank and well-directed response. If the meaning of "process" was restricted to processes that transformed matter, as the old "mental process" cases suggest and the Patent Office has repeatedly asserted, all of the software process patents earlier approved by the CCPA would seem to be invalid because all of them relate to the kinds of mental processes (namely, data

---

Benson, 441 F.2d at 686-87 (noting the "remarkable similarity" between the arguments made by the Patent Office in this case and in others, such as *Musgrave*, which the CCPA had rejected); *Foster*, 438 F.2d at 1014-15 (C.C.P.A. 1971) (criticizing the Patent Office for rejecting Foster's claims given its ruling in *Musgrave*).

[67] Although in all eight cases the CCPA overturned the Patent Office's rejection of the software process claims on subject matter grounds, the CCPA sometimes ultimately affirmed the Patent Office rejections on other grounds. Twice they ruled that the patent applicant had failed to satisfy the patent claim requirements because the applicant had not limited the process claims to machine implementations of the process. *See In re* Foster, 438 F.2d 1011 (C.C.P.A. 1971); *In re* Prater, 415 F.2d 1393 (C.C.P.A. 1969); *see also In re* Ghiron, 442 F.2d 985 (C.C.P.A. 1971) (claimant failed to disclose which computers could be used to implement his process); *In re* Bernhart, 417 F.2d 1395 (C.C.P.A. 1969) (striking some claims as obvious).

Curiously enough, one of the cases in which the CCPA affirmed denial of the patent based on a failure to limit the claims to machine implementations was decided after *Musgrave*. *See In re* Foster, 438 F.2d 1011 (C.C.P.A. 1971). The CCPA cited *Musgrave* approvingly in *Foster* as to the statutory subject matter issue in the case, but followed *Prater II* as to the need to claim machine implementation to satisfy section 112's requirements.

It was not until *Benson*, 441 F.2d 682, that the CCPA clearly broke away from requiring language in the claims that limited the process to machine implementations. (For discussion of the *Benson* claims, see *infra* notes 71-76 and accompanying text). Even then, the CCPA emphasized that Benson's process had no practical use except through machine implementations. *See Benson*, 441 F.2d at 688; *see also In re* McIlroy, 442 F.2d 1397, 1398 (C.C.P.A. 1971).

Given its apparent repudiation of the "mental process" doctrine in the *Musgrave* case, the CCPA's continuing emphasis on machine implementations to bolster its holdings was a curious thing about these decisions, perhaps reflecting some lingering doubt about the truly radical assertion that all processes that could be carried out on a computer were patentable.

[68] One important question not answered by the court in *Musgrave* was whether Musgrave's process was in the technological arts either because it was to be implemented by machine or because it related to the technological art of seismography. This is an important issue since how broadly the *Musgrave* case should be interpreted depends on this point. *Benson* can be construed as answering this question in favor of the former interpretation. This issue and related ones are discussed at length *infra* notes 315-45, 397-409, and accompanying texts.

representation, collection, and analysis) previously held unpatentable.[69] This, then, is the backdrop to the Supreme Court's decision in *Gottschalk v. Benson*.[70]

III. *GOTTSCHALK V. BENSON*: PROGRAM ALGORITHMS ARE NOT PATENTABLE

A. *Benson's Claims*

Benson made two claims for his new method of converting binary coded decimals to pure binary form.[71] Claim 8 described the method as it ap-

---

[69] All eight of the pre-*Benson* cases involved processes that could be performed (or at least simulated) by a human being without the aid of a machine, that is, either done in a person's head or with the aid of pen and paper. Because of the existence of general purpose digital computers, such processes could now be performed by computer as well. Although the Patent Office treated all eight cases as "mental process" cases, the four equation-related cases (*Prater*, *Bernhart*, *Musgrave*, and *Foster*) might as easily (and possibly more appropriately) have been characterized as "improved data analysis method" cases. Two of the other four cases (*Mahony* and *Benson*) might have been characterized as "improved data representation" cases, and the remaining two cases (*McIlroy* and *Waldbaum*) as "improved methods of identifying data" cases. These alternative characterizations of the cases come closer to the heart of what the author thinks that the "mental process" doctrine was intended to cover. *See supra* note 34.

[70] The CCPA gave various justifications for patentability of computer program-related inventions in the period before *Benson*. In view of the developments in later cases, each can be seen as a "false start." The first justification was that such processes are patentable as long as the claims were limited to machine implementations. *See supra* note 63 and accompanying text. A second was that such processes are patentable because each new program creates a new machine (or a new structure for a machine). *See supra* note 62. A third was that such processes are patentable because they are in the technological arts. *See supra* notes 66-67 and accompanying text. A fourth was that such processes are patentable because they improve the internal operation of the machines in which they operate. *See infra* note 88. (The third and fourth of these positions were later relied upon by the Patent Office as potentially limiting principles on the extent of patentability of programs, but the CCPA rejected the idea that these doctrines represented limiting principles. *See, e.g., In re* Toma, 575 F.2d 872 (C.C.P.A. 1978).) A fifth was that such processes are patentable if the only practical implementation of them is by computer. *See infra* note 88 and accompanying text.

Because all of these approaches are inconsistent with the Supreme Court's decision in *Benson*, they must be regarded as "false starts."

[71] For conciseness, there is no reference in the text to Arthur Tabbot, Benson's co-inventor. Both Tabbot and Benson were employees of AT&T Bell Laboratories, which was the assignee of the invention.

For those who do not immediately comprehend the nature of the Benson invention, a very brief explanation follows. To process numbers in a general purpose digital computer, it is necessary to convert the numbers from the standard Arabic representation which humans tend to use (*e.g.*, 53) to pure binary form (which can be expressed in human readable form as a sequence of "1's" and "0's") so that they can be represented electronically as a string of high and low voltages which correspond to

plied to the conversion of "signals"[72] and Claim 13 as it applied to the conversion of "representations."[73] In view of the manner in which similar claims had been interpreted in prior CCPA cases, Benson seemed in Claim 8 to seek a patent for machine implementations of his process,[74]

---

the "1's" and "0's" of the written binary representation of the numbers. A digital electronic computer can only "read" (*i.e.*, process and execute) the electronic representation.

As an intermediate step in the process of performing a such a conversion, numbers are represented in binary coded decimals (BCD). As an example, "53" is first broken down to a "5" in the 10 category and a "3" in the 1 category. Each of these digits is then represented in binary form: 0101 for the "5" and 0011 for the "3". The BCD form of 53 is 0101 0011. The BCD version of the number is converted to pure binary form more readily than the Arabic representation could be. Benson discovered a new way to perform the second stage conversion from BCD to pure binary form.

[72] Claim 8 read:

> The method of converting signals from binary coded decimal form into binary which comprises the steps of
> (1) storing the binary coded decimal signals in a reentrant shift register,
> (2) shifting the signals to the right by at least three places, until there is a binary "1" in the second position of said register,
> (3) masking out said binary "1" in said second position of said register,
> (4) adding a binary "1" to the first position of said register,
> (5) shifting the signals to the left by two positions,
> (6) adding a "1" to said first position, and
> (7) shifting the signals to the right by at least three positions in preparation for a succeeding binary "1" in the second position of said register.

Benson, 441 F.2d at 683.

[73] Although the language of the two claims differs more than in this respect, the CCPA accepts this distinction between Claims 8 and 13. *Id.* at 686-88.

Claim 13 read:

> A data processing method for converting binary coded decimal number representations into binary number representations comprising the steps of
> (1) testing each binary digit position i, beginning with the least significant binary digit position, of the most significant decimal digit representation for a binary "0" or a binary "1";
> (2) if a binary "0" is detected, repeating step (1) for the next least significant binary digit position of said most significant decimal digit representation;
> (3) if a binary "1" is detected, adding a binary "1" at the (i + 1)th and (i+3)th least significant binary digit positions of the next lesser significant decimal digit representation, and repeating step (1) for the next least significant binary digit position of said most significant decimal digit representation;
> (4) upon exhausting the binary digit positions of said most significant decimal digit representation, repeating steps (1) through (3) for the next lesser significant decimal digit representation as modified by the previous execution of steps (1) through (3); and
> (5) repeating steps (1) through (4) until the second least significant decimal digit representation has been so processed.

Id. at 683-84. Professor Chisum demonstrates this method in his article on *Benson*. *See* Chisum, *supra* note 12, at 973-74 n.48.

[74] *See, e.g., In re* Mahony, 421 F.2d 742, 745-46 (C.C.P.A. 1970) (interpreting "signals" as

whereas Claim 13 contained no language limiting the reach of the claim to machine implementations.[75] Benson asserted that Claim 13 was as patentable as Claim 8 because all that *Musgrave* required was that the process recited be in the "technological arts."[76]

Prior to Benson's discovery, there had been a number of known ways to convert binary coded decimals to pure binary form. Benson claimed to have invented:

> a better and simpler way of doing it having various advantages recited in the specification such as reducing the number of steps required to be taken, dispensing with the repetitive storing and retrieval of partially converted information, eliminating the need for interchanging signals among various equipment components and the need for auxiliary equipment, and decreasing the chance of error.[77]

B. *The Patent Office's Rejection of the Benson Claims*

Without contesting the novelty, utility, and inventiveness of the Benson discovery,[78] the Patent Board of Appeals upheld the Patent Office examiner's rejection of both claims, saying: "Claims 8 and 13 stand rejected as for subject matter not embraced by 35 U.S.C. section 101 in that they set forth 'mental processes' and 'mathematical steps,' neither being an 'art' as construed by a long line of decisions."[79] Benson argued that if the Board construed the claims in light of the specification as a whole, it would be

---

limiting a program-related process claim to machine implementations).

[75] *See supra* note 73 and accompanying text.

[76] The *Musgrave* standard can be found *supra* note 66 and accompanying text.

[77] *Benson*, 441 F.2d at 683.

[78] There was, however, some discussion in the CCPA opinion of whether the Board of Appeals had added a new ground of rejection based on failure to comply with section 112's requirement to make a proper specification of the invention (*i.e.*, construing the failure to limit method claims to machine implementations as making them overbroad and indistinct). *Id.* at 684-86. The CCPA rejected the section 112 argument. It attempted to distinguish the *Benson* case from *Prater II*, 415 F.2d 1393 (C.C.P.A. 1969), on the ground that Prater had admitted that his claim "read on" a mental implementation, whereas Benson had made no such admission (even though the CCPA itself clearly acknowledged that the claims covered non-machine implementations). *See infra* note 84 and accompanying text; *see also Benson*, 441 F.2d at 686. The CCPA also relied on *Musgrave* to uphold Benson's application on section 112 grounds. *Id.* This reliance occurred despite the fact that in *In re Foster*, which occurred between *Musgrave* and *Benson*, the CCPA had once again upheld the denial of a patent for failure to limit the claim language to machine implementations. *See supra* note 67.

[79] *Id.* at 684 (quoting the Board opinion).

clear that he was not claiming "mental processes." However, the Board of Appeals was not persuaded by this argument.[80]

C. *The CCPA Opinion on Benson's Claims*

The CCPA said it recognized that the *Benson* case differed from its other computer program-related decisions in that the other cases had generally involved some "subsidiary or additional art"[81] while the Benson claims were "directed solely to the art of data processing."[82] The CCPA did not regard this distinction as justifying a different conclusion about whether statutory subject matter had been claimed in this case.

In reviewing Claim 8, the CCPA pointed to the claim's reference to "signals" and "reentrant shift registers" (the latter a computer hardware component) to show that this claim did not cover a "mental process."[83] The CCPA went on to say: "The solicitor would have us hold the method is not a 'process' within section 101 on the ground that a programmable computer is merely a 'tool of the mind' and the method is basically

---

[80] *Id.* at 684-85.

[81] *Id.* at 686. The CCPA gave as examples the facts that *Bernhart* involved a planar illustration method and apparatus, *Mahony* involved a method of locating framing bits in a bit stream, and *Musgrave* and *Foster* involved seismographic analysis.

[82] *Id.* It strains matters to say that *Bernhart* (method for making two-dimensional representations of three-dimensional objects) and *Mahony* (method for framing bits to separate character strings) involved "subsidiary arts" in the same way that *Musgrave* and *Foster* did (seismic analysis). Indeed, the CCPA noted that in *Mahony* "we were concerned with data processing itself." *Id.* Although noting that Benson's invention related to "the art of data processing itself," the CCPA made nothing further of this point and emphasized instead the mechanical potential of the invention. *Id.* at 686-88.

[83] The reference to "reentrant shift register" was the most significant one. *See* Claim 8, *supra* note 72. The CCPA quoted an encyclopedia definition of "reentrant shift register," showing it was a hardware component, and emphasized that this term appeared in five of the steps of the process and by implication in a sixth. The CCPA taunted the patent solicitor for having to "admit" during oral argument that Claim 8's reference to this equipment "was, for him, its 'most embarrassing phrase.' " *Benson*, 441 F.2d at 687.

The CCPA then went on to say that Claim 8's reference to "signals," when read in light of a "reasonable interpretation" of the specification, "can only mean signals of the kind upon which the disclosed electronic digital computer hardware operates." It likened this term to the "bit stream" that the CCPA in *Mahony* had interpreted to be limited to machine implementations. *Id.*; *see In re* Mahony, 421 F.2d 742 (C.C.P.A. 1970). Had the CCPA been more consistent with another of its decisions *In re* Foster, 438 F.2d 1011 (C.C.P.A. 1971), rendered only months before *Benson*, it would have been forced to admit that in *Foster*, the CCPA had construed the term "signal" as not necessarily limited to machine implementations.

'mental' in character, apparently because the 'workstuff' of the method is numbers which are mathematical abstractions."[84] The CCPA did not find this persuasive, pointing out that cash registers also work with numbers, yet they are patentable.[85]

In examining Claim 13, the CCPA admitted that the claim "could in theory be [implemented by] any kind of writing implement and any kind of recording medium — 'pencil and paper' — or even, we suppose, red and blue poker chips and a surface to put them on or slots to put them in so that '0's' and '1's' can be represented."[86] Despite this, the CCPA opined that "in actual practice it seems improbable anyone would ever do that, speed measured in milli- or even microseconds being essential in the practical utilization of such a process."[87] Because the CCPA was convinced that the only practical use of this process was in connection with a digital electronic computer and because computers were an important technology, part of the "useful arts" rather than the "liberal arts," the CCPA was convinced that Claim 13, like Claim 8, was for patentable subject matter.[88]

---

[84] *Benson*, 441 F.2d at 687. While the many references to "reentrant shift registers" do seem to limit Claim 8 to machine implementations, it was in character for the CCPA to reject out of hand the analogy between the computer and the human mind which is so central to the "art" of artificial intelligence. *See supra* note 60, *infra* notes 391-94, and accompanying texts.

[85] *Benson*, 441 F.2d at 687. The point that the CCPA seemed to have missed from its own analogy is that regardless of whether adding machines or cash registers are patentable as machines, it is questionable whether a newly invented method of addition would be patentable as a process under section 101 of the patent statute. See *infra* notes 326-45 and accompanying text for a discussion of this issue.

[86] *Benson*, 441 F.2d at 688. Although the CCPA in *Benson* had distinguished between *Benson* and its previous cases on the ground that they had involved "subsidiary arts" whereas *Benson* did not, the CCPA ultimately ruled that, under *Musgrave*, it was not necessary for the claims to relate to such a subsidiary art to be patentable. This would seem to mean that as long as the process *could be* performed on a machine, such as a computer, it would be technological enough to be patentable. For further discussion of problems related to determining the proper interpretations of *Benson* and *Musgrave*, see *infra* notes 325-45, 397-409, and accompanying texts.

[87] *Benson*, 441 F.2d at 688.

[88] *Id.* The CCPA concluded its discussion of Claim 13 with a rhetorical flourish: "How can it be said that a process having no practical value other than enhancing the internal operation of [computing] machines is not likewise in the technological or useful arts?" *Id.*

D. The Supreme Court's Ruling in Benson

The Supreme Court unanimously reversed the CCPA's holding in *Benson*.[89] Justice Douglas' opinion spoke for the Court.[90] In overturning the CCPA's ruling, the Court, like the Patent Office before it, made no distinction between Claims 8 and 13.[91] Although the Supreme Court's *Benson* decision is not a model of clarity,[92] it is, in the author's view, supported by patent law.[93]

A close reading of the *Benson* opinion suggests that one of the things disturbing the Court was that the claims were for a method of transform-

---

[89] Gottschalk v. Benson, 409 U.S. 63 (1972). Justices Stewart, Blackmun, and Powell took no part in this decision. The unanimity of the Court about the unpatentability of Benson's process and the Court's reaffirmance of *Benson's* holding in subsequent cases, see *supra* note 8, is worth noting here, in view of Professor Chisum's critique of the *Benson* decision that seems to blame the ruling on Justice Douglas' muddled mind and antipatent bias. *See* Chisum, *supra* note 12, at 971-92.

[90] The Court received briefs from 14 amici in the *Benson* case. The four amici who urged the Court to reverse the CCPA ruling were major computer manufacturers and a trade association representing these manufacturers: Burroughs Corp., Honeywell, Inc., International Business Machines Corp., and Computer Business Equipment Manufacturers Association. *Benson*, 409 U.S. at 63.

Ten amici urged the Court to affirm the CCPA's ruling in *Benson*. Three were patent bar associations; seven were from firms or associations which were apparently developing software (Iowa State University Research Foundation, Institutional Networks Corp., Whitlow Computer Systems, Inc., Applied Data Research, Inc., Computer Software Analysts, Inc., Association of Data Processing Service Organizations, and Mobil Oil Corp. (Mobil was especially interested in the patentability issue in *Benson* because of its interest in patents for improved seismic analysis methods, such as that involved in *Musgrave*)).

Some have sought to explain the position of the major computer companies that opposed Benson's patent as economic protectionism of the interests of sellers of the machines in which programs would operate to the detriment of the software industry. *See, e.g.*, H. HANNEMAN, *supra* note 37, at 35-36. However, this ignores the fact that at the time, the computer companies themselves were among the most active developers of software products. Furthermore, this explanation does not account for the recommendations of Johnson's Presidential Commission. *See supra* notes 39-41 and accompanying text.

[91] This aspect of the Court's ruling in *Benson* seems to be a rejection of the argument that computer program-related inventions are patentable subject matter because they are capable of being performed by machine. After the Supreme Court's *Benson* decision, the CCPA generally ceased emphasizing the potential for machine implementation as a basis for finding patentability of program-related inventions. *But see* Paine, Webber, Jackson & Curtis, Inc. v. Merrill Lynch, Pierce, Fenner & Smith, Inc., 564 F. Supp. 1358 (D. Del. 1983), discussed *infra* notes 378-86 and accompanying text.

[92] Professor Chisum describes the reasoning in *Benson* as "monstrously bad." Chisum, *supra* note 12, at 977-78. The title of the section of the Chisum article dealing with the Supreme Court's case refers to the *Benson* decision as "The Bull In the China Closet." *Id.* at 971.

[93] For a discussion of Chisum's thesis and the author's critique of it, see *infra* notes 296-345 and accompanying text.

ing data from one form of representation to another.[94] The Court also seemed to be disturbed by the fact that numbers (themselves mathematical representations) were being transformed.[95] This transformation was to take place through a series of mathematical steps. The Benson procedure as a whole was characterized quite correctly by the Court as an "algorithm."[96]

---

[94] Early in the opinion, for example, the Court quoted the patent application as claiming an invention related to " 'the *processing of data* by program and more particularly to the programmed *conversion of numerical information*' in general purpose digital computers." *Benson*, 409 U.S. at 64 (emphasis added).

A few paragraphs later, the Court described what digital computers are and how they operate on numerical data, noting that computers "operate[] on data expressed in digits, solving a problem by doing arithmetic as a person would do it by head and hand." *Id.* at 65. The Court's decision to characterize the computer's ability to operate on data as analogous to how humans would do it demonstrates that this aspect of the Patent Office's argument had an impact on the Court's thinking.

These and other references to data representation and data processing suggest that the Court was influenced in its decision by this factor. Although the Court did not cite to prior patent decisions that deny patent protection to methods of representing data, *see, e.g.*, Berardini v. Tocci, 190 F. 329 (S.D.N.Y. 1911) (coding system held unpatentable), *aff'd*, 200 F. 1021 (2d Cir. 1912), it may still have been disturbed by the data representation character of the claims.

[95] In addition to the references cited in the preceding note, the Court devoted three paragraphs to describing decimal, pure binary, and BCD forms of representation of numbers. *Benson*, 409 U.S. at 66-67. The Court noted that conversion of BCD to pure binary form "could be done mentally" through use of a table printed in the text of the opinion. *Id.* at 67. (This was the second time the Court made reference to human capability of doing such a thing. *See supra* note 94).

The Court went on to point out that the Benson method "varies the ordinary arithmetic steps a human would use by changing the order of the steps, . . . and by taking subtotals after each successive operation," though also noting that Benson's method could be carried out without a computer. *Benson*, 409 U.S. at 67. The Court even put into the text of its opinion a table showing decimal and pure binary representations of numbers. *Id.* at 66.

The author regards it as significant that the Court chose to devote so much attention to these various representations of numbers in the text of the opinion. This demonstrates that it was partly the characterization of the invention as a way to represent information that caused the invention to be troublesome to the Court.

[96] The only mention of an "algorithm" in the CCPA's *Benson* decision was in a footnote in which the CCPA observed that Benson had characterized his invention as a machine algorithm before the patent examiner. *See Benson*, 441 F.2d at 685 n.2. (Judge Rich later questioned whether the *Benson* case really involved an algorithm. *See In re* Christensen, 478 F.2d 1392, 1396 (C.C.P.A. 1973) (Rich, J., concurring), *overruled*, *In re* Taner, 681 F.2d 787 (C.C.P.A. 1982).

It was apparently the amicus briefs of the computer companies opposed to software patents that focused the analysis on the patentability of "algorithms" and on the mathematical character of the Benson innovation. *See, e.g.*, Brief Amicus Curiae for Burroughs Corp. at 2, Gottschalk v. Benson, 409 U.S. 63 (1972) (No. 71-485) [hereinafter Burroughs Brief].

The algorithm argument seems to have greatly impressed the Court, for the Court abandoned the terms that the Patent Office, Benson's attorney, and the CCPA had been using to analyze the issue and fastened onto the characterization of Benson's method as an "algorithm." *Benson*, 409 U.S. at 65.

One of the arguments made to the Court in opposition to the grant of patent protection to algorithms was that innovative mathematical procedures had long been considered part of the "liberal arts" instead of the "useful arts," and hence, they were outside the scope of the patent system.[97] While the *Benson* opinion does not make reference to this view of mathematical innovations, the Court was persuaded by a different argument: Mathematical innovations should be treated like scientific truths and laws of nature, and scientific truths and laws of nature are unpatentable subject matter.[98] Part of the reason for the success of this argument, as later decisions made clear, was that the Court did not regard such things as human "inventions," but as "discoveries" of fundamental truths.[99]

Also affecting the Court's judgment about the Benson claims was the

---

The Court offered a definition of "algorithm" as "[a] procedure for solving a given type of mathematical problem." *Id.* This definition of algorithm becomes the focal point of the CCPA's effort to narrow the implications of the *Benson* decision to cases involving equations. *See infra* note 167 and accompanying text.

The Burroughs Brief contains an appendix displaying the Benson method in a more classically mathematical form than the claim language had. Burroughs Brief, *supra*, at Appendix A.

[97] Burroughs Brief, *supra* note 96, at 9-10.

[98] The Court quoted from four prior Supreme Court opinions for the proposition that ideas, scientific truths, laws of nature, and the like, are not patentable. *Benson*, 409 U.S. at 67. Nestled in between the third and fourth of these quotations is this interesting (if unexplained) sentence: "Phenomena of nature, though just discovered, *mental processes*, and abstract intellectual concepts are not patentable, as they are the basic tools of scientific and technological work." *Id.* (emphasis added). This seems to be a clear Supreme Court endorsement of the "mental process" doctrine of the patent law, a doctrine on which the Patent Office had long been relying to justify its position on computer program patents, and a doctrine which the CCPA had tried to repudiate. It is worth noting, however, that the Court did not seem to regard the "mental process" rule as dispositive on the patentability of inventions such as Benson's, for *this* is the *only* reference in the case to "mental processes."

[99] *See infra* notes 204-08, 272-75, and accompanying texts.

The "invention" versus "discovery" debate over mathematical insights is a longstanding one which has recently received attention in a book by Roger Penrose entitled *The Emperor's New Mind.* Here is some of what Penrose has to say on the subject:

> Is mathematics invention or discovery? When mathematicians come upon their results are they just producing elaborate mental constructions which have no actual reality, but whose power and elegance is sufficient simply to fool even their inventors into believing that these mere mental constructions are 'real'? Or are mathematicians really uncovering truths which are, in fact, already 'there' — truths whose existence is quite independent of the mathematicians' activities? I think that, by now, it must be quite clear to the reader that I am an adherent to the second, rather than the first, view, at least with regard to such structures as complex numbers and the Mandelbrot set.

R. PENROSE, THE EMPEROR'S NEW MIND: CONCERNING COMPUTERS, MINDS, AND THE LAWS OF PHYSICS 96 (1989).

breadth of the claims. The Court noted that the claims "were not limited to any particular art or technology, to any particular apparatus or machinery, or to any particular end use. They purported to cover any use of the claimed method in a general purpose digital computer."[100] This argument too had been made by the Patent Office and the amici.[101]

Seemingly, the most persuasive argument made to the Court was the argument that historically, "process" in the patent sense had been understood to mean processes that transformed matter.[102] Six long paragraphs of the Court's opinion are devoted to discussion of cases on this point.[103] Among the precedents upon which the Court relied was *Cochrane v. Deener*,[104] the case cited by the Patent Office for years to support its position that "mental processes" (and program-related processes that could be

---

[100] *Benson*, 409 U.S. at 67. The Court emphasized that Benson's process claims were "so abstract and sweeping as to cover both known and unknown uses of the BCD to pure binary conversion." *Id.* at 68. The Court gave examples of its range: "The end use may (1) vary from the operation of a train to verification of drivers' licenses to researching the law books for precedents and (2) be performed through any existing machinery or future-devised machinery or without any apparatus." *Id.*

The Court then moved into a discussion of O'Reilly v. Morse, 56 U.S. (15 How.) 62 (1854), a classic "overbreadth" case. Morse's first seven claims related to his invention of the telegraph. His famous eighth claim was for *all* uses of electromagnetism to communicate intelligible characters at a distance, regardless of the apparatus that might be used to implement the communication (although Morse had only invented one apparatus to implement it). The Court in *Morse* struck this overbroad claim as akin to a claim for a scientific principle, just as it was now doing in *Benson*. The *Benson* opinion contrasts *Morse* with The Telephone Cases, 126 U.S. 1 (1888), in which the Court ruled that Alexander Graham Bell had made claims, not for *all* telephonic uses of electricity (which would have been akin to Morse's overbroad eighth claim), but only for two methods which Bell had, in fact, invented. Thus, Bell's claims, unlike Morse's, were not overbroad. *Benson*, 409 U.S. at 68-69.

Professor Chisum properly attacks this part of the *Benson* decision as very confusing. *See* Chisum, *supra* note 12, at 984-85. Chisum also correctly points out that if an invention has a broad set of end uses, the patentee is generally regarded as entitled to reap the benefit of having made such a broadly useful contribution to the useful arts. *Id.* at 984-86. *But see infra* note 328 (concerning the breadth problems that might develop if someone sought a patent on a new method of performing addition).

[101] *See Benson*, 441 F.2d at 684; Burroughs Brief, *supra* note 96, at 5.

[102] The second paragraph of the Court's opinion poses the following question: "[W]hether the method described and claimed is a 'process' within the meaning of the Patent Act." *Benson*, 409 U.S. at 64. Because the Court reversed the CCPA's ruling in *Benson*, the Court quite obviously decided that Benson had not claimed a process that was patentable under the existing patent statute. Nevertheless, it is a curious fact — pointed out in the CCPA's first decision following *Benson*, *In re* Christensen, 478 F.2d 1392, 1395 (C.C.P.A. 1973) (Rich, J., concurring), *overruled*, *In re* Taner, 681 F.2d 787 (C.C.P.A. 1982) — that the Court did not directly answer its own question.

[103] *See Benson*, 409 U.S. at 69-71.

[104] 94 U.S. 780 (1877).

done by machine or "mentally") are not patentable.[105]

Despite its strong emphasis on "transformation" as the historical key to the patentability of processes, the Court denied that it was holding that "no process patent could ever qualify if it did not meet the requirements of our prior precedents."[106] The Court also denied that its decision in *Benson* would preclude issuance of any computer program patents[107] or "freeze process patents to old technologies, leaving no room for the revelations of the new, onrushing technology."[108]

As if anticipating that readers might be somewhat confused about its ruling, the Court concluded its discussion of Benson's case with a "nutshell" of its holding. After repeating that ideas cannot be patented, the Court asserted that the practical effect of a decision allowing Benson to patent his method of converting BCD numerals to pure binary form would be to grant a patent on an idea: "The mathematical formula involved here has no substantial practical application except in connection with a digital computer, which means that if the judgment below is affirmed, the patent would wholly pre-empt the mathematical formula and in practical effect would be a patent on the algorithm itself."[109]

---

[105] *See Benson*, 409 U.S. at 67; *see also supra* note 98. The Court even quoted *Cochrane*'s description of what "process" means in the patent sense. The pertinent part was as follows: "A process is a mode of treatment of certain materials to produce a given result. It is an act, or a series of acts, performed upon the subject-matter to be transformed and reduced to a different state or thing." *Benson*, 409 U.S. at 70 (quoting *Cochrane*, 94 U.S. at 788). This same passage is quoted, and transformation of matter emphasized strongly, in Diamond v. Diehr, 450 U.S. 175 (1981). *See infra* notes 265-68 and accompanying text. Professor Merges, in commenting on an earlier draft of this article, noted that the *Cochrane* definition of process may not, strictly speaking, require a transformation of matter. He notes that while a computer program does not transform matter, it does cause a computer to change from one "state" to another. While, if one took this interpretation of *Cochrane* seriously, programs themselves would be patentable, *Benson* and similar cases involve the *ideas* on which programs are based; these ideas do not directly act upon the computer or cause it to change state. *See* Dukarich, *Patentability of Dedicated Information Processors and Infringement Protection of Inventions that Use Them*, 29 JURIMETRICS J. 135 (1989) (discussing the patentable subject matter problems with inventive abstractions underlying programs).

[106] *Benson*, 409 U.S. at 71. There are two direct messages in this statement by the Court: first, that the Court perceived that the precedents to that point had limited the patentability of processes to those which transformed matter, and second, that the Court was not ruling out the possibility that, in the future, it might be willing to move beyond that standard. A third (but implicit) message of this statement was that Benson had not persuaded the Court that his case presented an appropriate instance to go beyond the traditional standard.

[107] *Id.*

[108] *Id.*

[109] *Benson*, 409 U.S. at 71-72. This "wholly pre-empt" language of the Court's "nutshell" be-

Although the Court said nothing else about Benson and his algorithm, the opinion concludes with a policy discussion of the patentability of computer programs, an issue which the amici and the Patent Office had urged the Court to address.[110] The Court did not feel competent to decide whether computer programs should be made patentable,[111] but pointed to the Presidential Commission that had "rejected the proposal that these programs be patentable."[112] After a lengthy quotation from the Commission's Report,[113] the Court urged Congress to take up the issue of whether computer programs should be patented.[114]

---

comes the focal point of the CCPA's effort to construe the *Benson* ruling as narrowly as possible. *See infra* notes 181-83 and accompanying text.

[110] *See, e.g.*, Burroughs Brief, *supra* note 96, at 24. The Burroughs Brief argues that computer programs were already adequately protected as an intellectual property by copyright law and also that copyright was a more appropriate form of intellectual property protection for programs than patent law because copyright would give protection to the program while leaving the program's underlying mathematics available to all to use. *Id.* at 22-23; *see also* 1966 REPORT, *supra* note 3, at 13, quoted *infra* note 113 (arguing against patents for program innovations because of the availability of copyright).

[111] *Benson*, 409 U.S. at 72. Implicitly, the Court seemed to say that without legislation "making" programs patentable, programs would not be so.

[112] *Id.*

[113] The Court quoted all three paragraphs of the 1966 Report dealing with the patentability of computer programs:

> Uncertainty now exists as to whether the statute permits a valid patent to be granted on programs. Direct attempts to patent programs have been rejected on the ground of nonstatutory subject matter. Indirect attempts to obtain patents and avoid the rejection, by drafting claims as a process, or a machine or components thereof, programmed in a given manner rather than as a program itself, have confused the issue further and should not be permitted.
>
> The Patent Office now cannot examine applications for programs because of a lack of classification technique and the requisite search files. Even if these were available, reliable searches would not be feasible or economic because of the tremendous volume of prior art being generated. Without this search, the patenting of programs would be tantamount to mere registration and the presumption of validity would be all but nonexistent.
>
> It is noted that the creation of programs has undergone substantial and satisfactory growth in the absence of patent protection and that copyright protection for programs is presently available.

Id. (quoting 1966 REPORT, *supra* note 3, at 13).

This quotation from the Commission's report in *Benson* is the *only* reference in the entire set of patent cases on computer program-related innovations to the fact that programs were being copyrighted. It is also the only reference to an argument against patents for program innovations on the ground that programs were already adequately protected by copyright. This argument was made by the amici. *See supra* note 110.

[114] *Benson*, 409 U.S. at 73. The Supreme Court's decision in a later computer program case, Parker v. Flook, 437 U.S. 584 (1978), also urged Congressional action on computer program patents.

E. Observations About the Supreme Court's Benson *Decision*

There are a number of notable aspects of the Supreme Court's *Benson* decision. Besides calling into question the patentability of computer programs,[115] the Supreme Court's *Benson* decision is important because it contains the first discourse on "algorithms" as a focal point of analysis of the patentability of computer program-related inventions. The case law before *Benson* is silent on the patentability of algorithms;[116] after *Benson*, it is focused almost exclusively on algorithms.[117]

Although the Supreme Court did provide a definition of the term "algorithm" in *Benson*,[118] its failure to elaborate further on what it meant by this term[119] led to the CCPA's questioning of whether *Benson* itself involved an "algorithm."[120] The Court's failure to identify which prior CCPA decisions contained claims enough like Benson's that they too should have been held to be for unpatentable subject matter (and which, if any, of the CCPA's earlier decisions would have been upheld on subject matter grounds)[121] meant that the CCPA had less guidance than it needed

---

Flook, 437 U.S. at 595-96. Congress did take up the issue of the patentability of programs, but enacted no legislation about it. *See, e.g.*, Gemignani, *supra* note 62, at 295.

[115] The aspect of *Benson* that has received most attention is its apparent questioning of the patentability of all program inventions. *See, e.g.*, Soltysinski, *Computer Programs and Patent Law: A Comparative Study*, 3 RUTGERS J. COMPUTERS & L. 1, 2-3 (1973); Comment, *Computer Program Classification: A Limitation on Program Patentability as a Process*, 53 OR. L. REV. 501, 504 (1974). Since this issue has been dealt with elsewhere and the Court's subsequent decision in *Diehr* makes clear that some programs-related inventions are patentable, this article will only note this aspect of the case.

[116] *See supra* notes 57-59 and accompanying text.

[117] This is discussed at length in Part IV.

[118] "A procedure for solving a given type of mathematical problem is known as an 'algorithm.' " *Benson*, 409 U.S. at 65.

[119] To discern how the Patent Office's post-*Benson* interpretation of the term "algorithm" differed from the CCPA's, see, for example, *In re* Freeman, 573 F.2d 1237 (C.C.P.A. 1978) and *In re* Chatfield, 545 F.2d 152 (C.C.P.A. 1976).

[120] *See, e.g., In re* Christensen, 478 F.2d 1392, 1396 (C.C.P.A. 1973) (Rich, J., concurring) (expressing his disbelief that the Benson method was an "algorithm" as the Court had defined the term), *overruled, In re* Taner, 681 F.2d 787 (C.C.P.A. 1982).

[121] *See, e.g.*, Soltysinski, *supra* note 115, at 44-52 (concluding it was unclear how the Supreme Court's *Benson* decision would affect prior CCPA computer program patentability decisions).

Waldbaum is the one case (apart from *Benson* itself) about which it is possible to ascertain how the CCPA would view one of its pre-*Benson* cases after the Supreme Court's ruling in *Benson*, for the CCPA reversed its first ruling on the patentability of this invention after the Supreme Court's ruling in *Benson* and relied on *Benson* as a basis for the second ruling. *See Waldbaum I*, 457 F.2d 997 (C.C.P.A. 1972), *aff'd Waldbaum II*, 559 F.2d 611 (C.C.P.A. 1977). Curiously enough, *Waldbaum I*

in order to follow the Supreme Court's ruling.

It is a striking feature of the *Benson* opinion that, apart from reporting that the CCPA had upheld Benson's claims, the Court did not even mention a single one of the eight CCPA cases involving computer program-related claims. Nor does it cite any other CCPA decisions, or even any Patent Board of Appeals decisions. In fact, the only cases the Court either discussed or cited in the entire *Benson* opinion are its own prior decisions, and most of them were from the nineteenth century.

Although it may have smarted for the CCPA judges to have one of their opinions overturned by the Supreme Court, it must have smarted more not to have their decisions noticed at all. Of course, a decision that did not deign to address and refute the arguments that the CCPA judges had found so persuasive was not likely to be regarded as instructive and persuasive in subsequent cases that would come before the CCPA. On the theory that they could not exactly determine what the Court's holding was in *Benson*, it was natural for the CCPA judges to fasten upon the limiting language of the *Benson* opinion, such as "algorithm" and "wholly preempt," and the Court's observation that *Benson*'s claims were not limited to particular machines or domains. As a result, they could interpret *Benson* as rejecting only this one set of process claims and not as a criticism of the line of cases the CCPA had developed.

However, if one searches through the *Benson* decision for evidence to determine which "side" of the computer program patentability debate the Court favored, one would certainly find much more in the opinion to support the Patent Office's position than to support the CCPA's position. For one thing, the Court specifically stated that "mental processes" are not patentable.[122] For another, the Court, like the Patent Office, referred to

---

was relied on for support in another controversial patentable subject matter case, *In re* Bergy, 563 F.2d 1031, 1038 (C.C.P.A. 1977), *vacated and remanded for reconsideration in light of Parker v Flook, sub nom.* Parker v. Bergy, 438 U.S. 902 (1978) (patentability of genetically engineered life forms), despite the fact that the CCPA had effectively overruled that decision only about two months earlier in *Waldbaum II. See In re* Bergy, 596 F.2d 952, 959 (C.C.P.A. 1979), *vacated and remanded for dismissal for mootness*, 444 U.S. 1028 (1980); *In re* Sarkar, 588 F.2d 1330, 1333 (C.C.P.A. 1978).

[122] *Benson*, 409 U.S. at 67. As will be shown later, the phrases "mental process" and "mental steps," while admittedly awkward and not entirely apt for describing some important patentable subject matter problems, may be closer to the heart of the problem with software patentability than the term "algorithm," which in the post-*Benson* case law comes to dominate the debate. *See infra* notes

Cochrane and other such cases as support for its position that processes like Benson's are not patentable.[123] The Court, like the Patent Office, made no distinction between Claims 8 and 13 and regarded both as unpatentable subject matter.[124] Moreover, the Court's characterization of digital computers as instrumentalities which operate on data in a manner similar to the way a human might by head or hand was essentially the same analogy as the patent solicitor had used before the CCPA.[125] The Supreme Court's concern over the sweeping and abstract nature of the claims in *Benson* is also found in the Patent Board of Appeals' opinion in the *Benson* case.[126]

Conversely, the Court made little use of arguments that appear in CCPA decisions. While the Court did seem to accept the CCPA's assertion that the only practical use for the Benson method was by machine, the Court drew the opposite legal conclusion from this premise than the CCPA had drawn.[127] On the other hand, the Court refused to announce a blanket rule that computer program-related inventions are unpatentable.[128] It did not seem to rely on the "mental process" doctrine as support for its ruling in *Benson*.[129] Furthermore, although noting that the

---

352-77 and accompanying text. As was shown earlier, the "mental process" cases were mainly aimed at preventing patent protection for data gathering and analysis. *See supra* notes 27-34. The Supreme Court in *Benson* made much of the fact that Benson's claim was for a "data processing" method. *See supra* notes 94-95. The Court perceived Benson's method of "converting" data from one form of representation to another as an algorithm, although it did not explicitly connect either the data processing or the algorithm issue to the mental process issue. In fact, both can be subsumed under the Patent Office's "mental process" objection to program patentability. *See infra* notes 352-77 and accompanying text.

[123] *Benson*, 409 U.S. at 69-71. In *Benson*, the Supreme Court referred to transformation of matter as an important factor in the patentability of a process three times. This characterization had also been relied upon by the Patent Office. *See supra* notes 46-47 and accompanying text.

[124] *See supra* note 91 and accompanying text.

[125] *See supra* notes 94-95 and accompanying text.

[126] *See Benson*, 441 F.2d at 684.

[127] The CCPA concluded that Claim 13 was patentable because, despite the fact that its language did not limit it to machine implementations, its only practical application was in a machine. *See id.* at 688. The Supreme Court, however, concluded that in view of the fact that its only practical application was in a computer, granting Benson's claim would wholly preempt use of the algorithm. *Benson*, 409 U.S. at 71-72. (This may explain why the Court did not distinguish between Claims 8 and 13.).

[128] *Benson*, 409 U.S. at 71.

[129] The Court mentioned "mental processes" as being among the processes that are not patentable. *Id.* at 67. However, the Court did not rely on this directly in ruling on Benson's application, nor did it directly describe Benson's process as a mental one.

meaning of the term "process" in the patent sense had been historically confined to processes that transformed matter, the Court explicitly stated that it did not mean that it would never move beyond this point. Benson, however, had not persuaded the Court to construe "process" more broadly in this case.

Seen in this light, the Supreme Court's *Benson* opinion, while not free from ambiguity, can be seen as an opinion consistent with patent decisions of the past and as one steering a middle course between the positions of the Patent Office and the CCPA in their struggle over the patentability of computer programs. By deciding to act on a case-by-case basis rather than by general pronouncements, the Court was steering a traditional course.

IV. FROM *BENSON* TO *DIEHR*: THE EVOLUTION OF THE CCPA'S ANALYSIS OF CLAIMS FOR ALGORITHMS AND OTHER PROGRAM-RELATED INVENTIONS

A. *Overview of the Rulings During This Period*

In the ten years between the Supreme Court's decisions in *Gottschalk v. Benson*[130] and *Diamond v. Diehr*,[131] there were twenty CCPA decisions on the patentability of computer program-related inventions on subject matter grounds.[132] The CCPA reversed Patent Office rejections in twelve of these cases and affirmed them in eight others. That eight of the Patent Office's rulings were upheld by the CCPA did not mean that the battle

---

[130] 409 U.S. 63 (1972).

[131] 450 U.S. 175 (1981).

[132] Two other computer program-related patent cases, decided during this period by courts other than the CCPA, contain some reference to patentability issues. *See* Arshal v. United States, 621 F.2d 421 (Ct. Cl. 1980) (invalidating on subject matter grounds a patent on a flight control system which the United States was said to have infringed); Hirschfield v. Banner, 462 F. Supp. 135 (D.D.C. 1978) (CCPA Judge Markey, sitting by designation, briefly discussed the patentability of a program-related invention in the course of an opinion on the adequacy of disclosure (*i.e.*, whether it would require undue experimentation to write a program to implement the system)).

Two other patent cases involving computer program-related innovations were decided during this period, but they contain no discussion of subject matter issues. *See* Decca Ltd. v. United States, 544 F.2d 1070 (Ct. Cl. 1976) (finding patent claims for radio navigation system valid and infringed), *rev'd in part*, 640 F.2d 1156 (Ct. Cl. 1980); Digitronics Corp. v. New York Racing Ass'n., 187 U.S.P.Q. (BNA) 602 (E.D.N.Y. 1975) (finding patent claims for a data processing system related to ticket issuing machines invalid and uninfringed), *aff'd*, 553 F.2d 740 (2d Cir.), *cert. denied sub nom.* Amperex Electronic Corp. v. New York Racing Ass'n, Inc., 434 U.S. 860 (1977).

between the Patent Office and the CCPA over the patentability of computer program-related innovations had begun to abate; rather, it only entered new rounds.[133]

Five main themes emerged in these twenty CCPA decisions on computer program-related inventions. The predominant theme was the CCPA's effort to find an interpretation of *Benson* that would greatly confine its application (in contrast to the Patent Office, which read *Benson* very broadly). The CCPA's most common interpretation was that *Benson* forbade only patents on "mathematical algorithms" (by which it generally meant equations), and then only when granting the patent would "wholly pre-empt" use of the algorithm (*i.e.*, when *any* use of the equation would infringe the patent).[134] A legal distinction between mathematical and nonmathematical algorithms is noteworthy mainly because computer scientists and software developers would regard it as a false distinction: All computer program algorithms are mathematical in nature.[135]

A second theme was the CCPA's clear refusal to entertain any arguments directed at finding additional subject matter limitations on the patentability of program-related innovations besides the "algorithm" rule it found in *Benson*. When faced, for example, with challenges to the patentability of data processing programs (such as a financial record keeping system or a natural language translation process), the CCPA swept aside doubts about patentability by finding that since no "mathematical algorithms" (that is, equations) were recited in the claims, *Benson* was inapplicable and the innovation was, therefore, patentable.[136]

---

[133] Because of its continuing disagreement with the CCPA over the proper interpretation of *Benson*, the Patent Office petitioned for certiorari in seven of these 20 cases. The Court granted four petitions, denied two others, and dismissed one for untimely filing. *See In re* Sherwood, 613 F.2d 809 (C.C.P.A. 1980), *cert. denied*, 450 U.S. 994 (1981); *In re* Diehr, 602 F.2d 982 (C.C.P.A. 1979), *aff'd sub nom.* Diamond v. Diehr, 450 U.S. 175 (1981); *In re* Bradley, 600 F.2d 807 (C.C.P.A. 1979), *aff'd court sub nom.* Diamond v. Bradley, 450 U.S. 381 (1981) (by equally divided court); *In re* Flook, 559 F.2d 21 (C.C.P.A. 1977), *rev'd sub nom.* Parker v. Flook, 437 U.S. 584 (1978); *In re* Chatfield, 545 F.2d 152 (C.C.P.A. 1976), *cert. dismissed*, 434 U.S. 875 (1977) (untimely filing); *In re* Noll, 545 F.2d 141 (C.C.P.A. 1976), *cert. denied*, 434 U.S. 875 (1977); *In re* Johnston, 502 F.2d 765 (C.C.P.A. 1974), *rev'd sub nom.* Dann v. Johnston, 425 U.S. 219 (1976) (reversing on obviousness grounds, but not reaching the subject matter issue).

[134] *See, e.g., In re* Freeman, 573 F.2d 1237 (C.C.P.A. 1978) (discussed at length *infra* notes 178-88 and accompanying text).

[135] *See infra* notes 389-94 and accompanying text.

[136] *See, e.g., In re* Toma, 575 F.2d 872 (C.C.P.A. 1978); *In re* Johnston, 502 F.2d 765 (C.C.P.A. 1974), *rev'd on other grounds sub nom.* Dann v. Johnston, 425 U.S. 219 (1976). But, for

A third theme which enlivened this period was the battle between the Patent Office and the CCPA over use of a "point of novelty" test in judging the patentability of program-related inventions.[137] If the algorithm was the only thing that distinguished the process from the prior art, the Patent Office considered the claim to be for the algorithm and hence, to be unpatentable under *Benson*. Although the CCPA itself seemed to endorse this test in its first program-related decision after *Benson*,[138] the CCPA soon repudiated it.[139] This repudiation, however, did not deter the Patent Office from continuing to use the test. The Supreme Court, while not explicitly endorsing the "point of novelty" test in *Parker v. Flook*,[140] nevertheless ruled in a manner consistent with use of this test. Undeterred by the Supreme Court's *Flook* decision, the CCPA continued to attack the "point of novelty" test, asserting that it improperly imported into the subject matter determination (a section 101 issue) considerations of novelty (section 102) and nonobviousness (section 103) issues. In *Diamond v. Diehr*,[141] the Supreme Court adopted the CCPA's alternative "subject matter as a whole" test and put the "point of novelty" debate to rest.

A fourth theme that emerged in the post-*Benson* period was a concern that computer program-related claims needed to be examined carefully so that *Benson*'s proscriptions against the patenting of algorithms could not be circumvented by "artful" claim drafting. This concern was the main reason for the Patent Office's use of the "point of novelty" test, for the Office thought that without the filter that this test provided, patent applicants would try to get around the rule against patenting algorithms by "tacking on" to the claims some conventional pre- or post-solution activity in order to bypass the subject matter hurdle. Although chiefly the Patent Office and Supreme Court worried about this problem,[142] even the CCPA, on occasion, intoned against such evasion, particularly when the evasion was sought to be accomplished by drafting the claim in apparatus rather

---

a discussion of more recent cases suggesting that, at least on occasion, the CCPA and its successor court are open to a broader interpretation of *Benson*, see *infra* notes 363-77 and accompanying text.

[137] This controversy is discussed at length *infra* notes 189-208 and accompanying text.

[138] *In re* Christensen, 478 F.2d 1392 (C.C.P.A. 1973), *overruled*, *In re* Taner, 681 F.2d 787 (C.C.P.A. 1982) (discussed *infra* notes 191-208 and accompanying text).

[139] *See, e.g.*, *In re* Chatfield, 545 F.2d 152, 158 (C.C.P.A. 1976).

[140] 437 U.S. 584 (1978); *see infra* notes 193-215 and accompanying text.

[141] 450 U.S. 175 (1981); *see infra* notes 260-81 and accompanying text.

[142] *See, e.g.*, Parker v. Flook, 437 U.S. 584 (1978), discussed *infra* notes 193-215 and accompanying text.

than method form.[143]

A fifth theme became more evident toward the end of the span between *Benson* and *Diehr* when the CCPA was seeking to strengthen its position after the Supreme Court in *Flook* had, for the third time in six years,[144] reversed one of its rulings in favor of a patent for program-related claims. This theme was an increasing emphasis on the industrial nature of processes for which computer programs were claimed as an element. Had Diehr's rubber curing process not been traditionally industrial in nature, the Supreme Court might well have ruled the other way in *Diehr*.[145]

It is no exaggeration to say that during the years after *Benson*, the CCPA studiously ignored most of what the Court had said in *Benson*, as well as in *Flook*, and struggled mightily to find ways to construe *Benson* as narrowly as possible. This does not mean, however, that the Supreme Court's actions had no influence in shaping the way the CCPA reacted to cases involving the patentability of computer program-related inventions. Just how the Court's decisions affected the CCPA's analysis of program-related claims is set forth in three subsections below, because each of the three phases of the time period between *Benson* and *Diehr*, began and ended with a Supreme Court-related action.

*B. Dissension in the CCPA in the Initial Post-*Benson *Phase*

In the first four years after the Supreme Court's decision in *Gottschalk v. Benson*,[146] the CCPA reviewed four cases involving computer program-related inventions for which patent applications had been denied by the Patent Office. Only in the first case, *In re Christensen*, did the CCPA agree with the Patent Office that the claims did not recite patentable subject matter.[147] Moreover, *Christensen* is the only case in which the CCPA

---

[143] *See, e.g., In re* Freeman, 573 F.2d 1237, 1247 (C.C.P.A. 1978); *see also In re* Richman, 563 F.2d 1026, 1030 (C.C.P.A. 1977) (expressing concern about evasive claim drafting). But, for discussion of the CCPA's early interpretation of *Benson* as applying only to "method" claims, see *infra* note 163 and accompanying text.

[144] Between *Benson* and *Flook*, the Supreme Court reversed the CCPA's ruling in *In re* Johnston, 502 F.2d 765 (C.C.P.A. 1974), *rev'd sub nom.* Dann v. Johnston, 425 U.S. 219 (1976). *Johnston* is discussed *infra* notes 155-60 and accompanying text.

[145] 450 U.S. 175 (1981); *see infra* notes 247, 260-81, and accompanying texts.

[146] 409 U.S. 63 (1972).

[147] 478 F.2d 1392 (C.C.P.A. 1973), *overruled, In re* Taner, 681 F.2d 787 (C.C.P.A. 1982)

judges were unanimous on the patentable subject matter issue. Even so, Judge Rich's concurrence in *Christensen* revealed some dissension in the CCPA over analyzing program-related claims.[148] The other three CCPA rulings in the initial post-*Benson* period overturned the Patent Office rejections of program-related claims, but all were by three-to-two votes[149] and generally reflected disagreement among the CCPA judges over the interpretation that should be given to *Benson*.[150] Because each of these decisions helped to shape the evolution of the law on the patentability of program-related inventions, each is briefly discussed below.

The primary significance of the CCPA's first post-*Benson* decision, *In re Christensen*, was in providing the Patent Office with a new ground for rejecting program-related claims.[151] The CCPA affirmed the rejection of Christensen's claims for an improved method of seismographic analysis, which was said to make it possible to obtain a continuous plotting of the porosity of subsurface formations. The CCPA majority characterized the issue in the case to be whether "a method claim in which the point of

---

(*Christensen* was not really overruled; *see infra* note 198).

[148] *See id.* at 1395-96 (Rich, J., concurring). While most of Judge Rich's criticism in the concurrence to *Christensen* was aimed at the Supreme Court's *Benson* decision, he not only made clear his reluctance about concurring with his colleagues in the result, but took indirect aim at the "point of novelty" language in the majority opinion.

Judge Rich (author of both the *Musgrave* and *Benson* opinions for the CCPA, and therefore the judge whose views on the patentability of such inventions had been rejected most directly by the Court in *Benson*) demonstrated that he had not been persuaded by the Supreme Court's *Benson* decision that Benson's innovation was an unpatentable process. He criticized the Court for failing to discuss what "process" means in the patent statute and described the Court's rationale in that case as a "mystery." *Id.* He agreed, however, that under the Supreme Court's *Benson* decision, the CCPA had no choice but to deny a patent on Christensen's claims because the "reasoning in the Supreme Court's opinion has more bearing on the facts of this case than it had on the facts before it in *Benson*. The claims in this case do contain a mathematical formula; in *Benson* they did not." *Id.* at 1396. Judge Rich found in the *Benson* "nutshell" the rule "that patents are not to issue where their effect would be to enable the patentee to prevent others from making use of a mathematical formula. That would be the effect here." *Id.*

[149] *In re* Chatfield, 545 F.2d 152 (C.C.P.A. 1976) (Lane & Rich, JJ., dissenting), *cert. dismissed for untimely filing*, 434 U.S. 875 (1977); *In re* Noll, 545 F.2d 141 (C.C.P.A. 1976) (Lane & Rich, JJ., dissenting), *cert. denied*, 434 U.S. 875 (1977); *In re* Johnston, 502 F.2d 765 (C.C.P.A. 1974) (Rich, J., and Markey, C.J., dissenting), *rev'd sub nom.* Dann v. Johnston, 425 U.S. 219 (1976).

[150] Judge Markey's dissent in *Johnston* was the only dissent in this period not based on subject matter grounds. Judge Rich's dissent in *Johnston* is discussed *infra* notes 158, 161. His dissent in *Chatfield* (in which Judge Lane joined) is discussed *infra* note 166. Judge Lane's dissent in *Noll* (in which Judge Rich joined) is discussed *infra* note 163.

[151] *Christensen*, 478 F.2d 1392.

novelty is a mathematical equation to be solved as the final step of the method, [was] a statutory method."[152] It held that *Benson* precluded a patent for such an invention.[153] Thereafter, whenever the "point of novelty" in a claim was an equation or a method of calculation, the Patent Office rejected the claim. The dispute between the CCPA and Patent Office over use of this test for patentability came to a head in both the *Flook* case during the second post-*Benson* phase and in the *Diehr* case in the third.[154]

In the year after *Christensen*, the CCPA was presented with another computer program-related case, *In re Johnston*.[155] Johnston made apparatus claims for a system of computerized analysis of financial records.[156] The Patent Board of Appeals had rejected the claims on the grounds that they were for a computer program (unpatentable subject matter under *Benson*), that they were for an unpatentable business method, and that they were too obvious to qualify for a patent.[157] The CCPA majority re-

---

[152] *Id.* at 1394.

[153] Under *In re* Musgrave, 431 F.2d 882 (C.C.P.A. 1970), itself a case involving equations for an improved method of seismographic analysis, Christensen's invention would unquestionably have been found patentable as within the "technological arts." *See supra* note 66 and accompanying text.

Judge Lane explained why *Benson* required rejection of the Christensen claims: "Given that the method of solving a mathematical equation may not be the subject of patent protection, it follows that the addition of old and necessary antecedent steps of establishing values for the variables in the equation cannot convert the unpatentable method into patentable subject matter." *Christensen*, 478 F.2d at 1394.

[154] The controversy about the use of a "point of novelty" test is discussed *infra* notes 189-208 and accompanying text. Because use of the "point of novelty" approach was later rejected by the CCPA, and ultimately by the U.S. Supreme Court, it is fair to characterize its use in *Christensen* as the first "false start" in the post-*Benson* era. In subsequent cases, most notably *In re* Flook, 559 F.2d 21 (C.C.P.A. 1977), *rev'd sub nom.* Parker v. Flook, 437 U.S. 584 (1978), the CCPA reinterpreted *Christensen* as a case in which patent protection had been denied only because the "last step" in his process was a calculation, not because the "point of novelty" of his invention was an equation.

[155] 502 F.2d 765 (C.C.P.A. 1974), *rev'd sub nom.* Dann v. Johnston, 425 U.S. 219 (1976). The patent examiner in this case had rejected the claims under section 102 (for lack of novelty) and section 112 (for indefiniteness). It was the Board of Appeals which raised questions about the claims under section 101 (inappropriate subject matter) and section 103 (for obviousness). The Board agreed with the examiner on the section 112 issue.

[156] Johnston had originally made process claims related to this invention as well, but chose to appeal only the apparatus claims. *Johnston*, 502 F.2d at 768 n.6. His system involved marking bank checks and deposit slips with machine-readable encoded magnetic symbols representing different kinds of transactions so that statements could be prepared which would provide totals for different categories of receipts and expenditures. It was to be carried out largely by computer program. *See id.* at 765-67 (containing the CCPA's description of the invention).

[157] *Id.* at 769. In support of its "business method" rejection, the Patent Office cited *In re* Mus-

garded *Benson* as inapplicable because Johnston's claims were not for a mathematical formula. The majority also rejected the characterization of Johnston's claims as for a business method and said, rather, that they were for a "record keeping machine system."[158] The CCPA further concluded that Johnston had satisfied the nonobviousness standard.

The Supreme Court, finding the claimed invention to be obvious, unanimously reversed the CCPA ruling in *Johnston*.[159] In the course of explaining that it was unnecessary to reach the subject matter issue also raised in the CCPA's *Johnston* opinion, Justice Marshall made one reference to *Benson* as a "limited holding."[160] In the next two program-related patentability decisions, *In re Noll*[161] and *In re Chatfield*,[162] handed down

---

grave, 431 F.2d 882 (C.C.P.A. 1970), as limiting patents to the technological arts. The Board of Appeals regarded the "apparatus" claim as a "dress" for what was essentially a claim for a banking service. For a discussion of apparatus claims for a financial record keeping system for which patentability was upheld, see Paine, Webber, Jackson & Curtis, Inc. v. Merrill Lynch, Pierce, Fenner & Smith, Inc., 564 F. Supp. 1358 (D. Del. 1983), and *infra* notes 378-86 and accompanying text.

[158] *Johnston*, 502 F.2d at 771. Judge Rich dissented on the ground that *Benson* required reversal. While he was quite familiar with the argument that new programs make computers into new machines, he said that his problem was that "knowing the invention to be a new program, I must decide whether it is patentable in any claimed form in view of *Benson*, whether claimed as a machine, a machine system, or otherwise." *Id.* at 773. Judge Rich said that if one focused on what the Court *did* in *Benson* rather than on its confused explanation of its decision, it was more consistent with the "thrust" of the Court's decision to rule against the patentability of Johnston's claims as well. *Id.* at 774.

[159] Dann v. Johnston, 425 U.S. 219, 220 (1976). The Patent Board of Appeals had ruled that Johnston's system was an obvious variation on the bookkeeping systems already used by banks or on a previously patented system for industrial businesses. *Johnston*, 502 F.2d at 769. The CCPA majority distinguished the prior systems from Johnston's system and ruled that Johnston's invention was not obvious. *Id.* at 771. Judge Markey's dissent agrees with the Board of Appeals that Johnston's system was too obvious to be patented. *Id.* at 772. The Supreme Court agreed with both bases of the Patent Board's obviousness ruling. *See Johnston*, 425 U.S. at 225-29.

[160] *Johnston*, 425 U.S. at 224.

Although the Supreme Court did not rule on the patentable subject matter issue in *Johnston*, it does appear that the Court, like the Patent Board of Appeals, perceived the claimed invention to be a banking service, rather than a new machine. *Id.* at 221-23.

In view of the Court's decision in *Benson* not to make a general pronouncement that computer program-related inventions were all unpatentable, it is not surprising that the Court once again in *Johnston* refrained from making such a general pronouncement. That it could reach a unanimous decision on obviousness grounds meant a broad ruling of this sort was unnecessary. The Court probably also wanted to give the Patent Office and the CCPA a chance to develop the law about program-related inventions somewhat before it took another subject matter case. *Johnston*, after all, was only the second CCPA decision since *Benson* and only the first post-*Benson* case to uphold a program-related patent.

[161] *In re* Noll, 545 F.2d 141, 149 (C.C.P.A. 1976) (noting the significance of the fact that the

shortly after the Supreme Court's *Johnston* decision, a majority of the CCPA judges seized on this reference to *Benson* as an endorsement of the CCPA's narrow interpretation of that case. The primary significance of *Johnston*, then, is this perceived endorsement of the CCPA approach to analyzing program-related claims.

In *Noll*, as in *Johnston*, the CCPA seemed to indicate that *Benson*'s proscriptions could be avoided merely by claiming a program-related invention in "apparatus" rather than in "method" form, even though it required only minor wording changes to draft claims in one form or the other.[163] In both *Noll* and *Chatfield*, the CCPA majority construed *Benson* as only applying to claims "for" equations,[164] and even then, suggested that field of use limitations would save equation claims, as long as use of the equations would not be "wholly pre-empted" by the patent

---

Supreme Court in *Johnston* "emphasized that *Benson* was a *limited* holding," despite Judge Rich's dissent in *Johnston* asking the Court to clarify the meaning of its *Benson* decision) (emphasis in original), *cert. denied*, 434 U.S. 875 (1977).

[162] *In re* Chatfield, 545 F.2d 152, 156 (C.C.P.A. 1976) (indicating that whatever doubt might have existed about the implications of *Benson* for the patentability of computer programs was "fully removed by the Supreme Court's unanimous characterization of its holding in *Benson* as 'limited' in *Johnston*"), *cert. dismissed for untimely filing*, 434 U.S. 875 (1977).

[163] *See Noll*, 545 F.2d at 148-49 (apparatus claims for the display of information on a cathode ray tube, which differed from the prior art by its use of a computer program); *Johnston*, 502 F.2d at 771 (apparatus claims for record keeping system). In neither case did the CCPA majority explicitly say that *Benson* could be averted by such minor wording changes, but that is clearly what the dissenters thought the majority was doing. Judge Lane's dissent in *Noll*, for example, (in which Judge Rich joined) expressed the view that it made no sense to rule, as his colleagues had, that *Benson*'s proscriptions could be avoided merely by claiming inventions in "apparatus" form ("system for . . .") rather than "method" form ("method for . . . consisting of the following steps"). *Noll*, 545 F.2d at 152. Judge Lane's dissent in *Noll* also took issue with the majority's view that *Benson* had not questioned the patentability of computer programs:

> Although the holding of the Court in *Benson* may have been narrower than the [Court's] language suggests, thus perhaps relegating that quoted language to the status of dicta, nevertheless the dicta remain as an indication of the Court's thinking in this area, which we would be well advised to heed.

Id. at 151.

Because this distinction between method and apparatus claims (used as a way of avoiding an unpatentable subject matter ruling under *Benson*) is later explicitly rejected by the CCPA in *In re* Freeman, 573 F.2d 1237 (C.C.P.A. 1978), it is safe to call this distinction yet another "false start" by the CCPA in its program-related patentability decisions.

[164] *Chatfield*, 545 F.2d at 156 (holding that *Benson* precluded patents only where the claims if granted "would have preempted all practical use of both the underlying mathematical formula and the involved algorithm"); *see Noll*, 545 F.2d at 148 (finding *Benson* inapplicable because Noll's claims were not for "an abstract 'law of nature, a mathematical formula, or an algorithm' ") (quoting *Johnston*, 502 F.2d at 765).

grant.[165] The CCPA majority tried very hard in *Chatfield* to move the discourse on patentability away from the subject of computer programs[166] by pointing to the Court's definition of the term "algorithm" as an indication that the only patentability problems presented in program-related cases concerned efforts to patent mathematics.[167]

---

[165] *See Chatfield*, 545 F.2d at 156-59 (suggesting that *Benson* only applied when a patent on the claims would preempt *all* uses of a mathematical formula and mathematical algorithms); *Noll*, 545 F.2d at 148 (distinguishing *Noll* from *Benson* on the ground that Noll's claims were limited to a particular technology).

Because employing "end use" limitations as a way to save the patentability of program-related inventions was rejected in *Waldbaum II*, 559 F.2d 611 (C.C.P.A. 1977), *see infra* note 175 and accompanying text, it is fair to characterize this approach as yet another "false start" by the CCPA.

[166] *Chatfield*, 545 F.2d 152. Chatfield claimed an improved method of assigning priorities among programs in a multiprogramming environment of a digital computer. It involved periodic halting of the processor operation so that program use data could be analyzed. That analysis would then determine the priorities for resource access for the next period of operation. The net effect of the invention was said to be "an overall gain in operating efficiency, *i.e.*, an increased throughput." *Id.* at 154. Insisting that Chatfield's claim was not for a program, but only for a process, the CCPA stated:

> Because nothing in the statute or in prior judicial opinions either authorizes or precludes patent protection of 'computer programs,' the mere labeling of an invention as a 'computer program' does not aid in decision making. . . . The Supreme Court having expressly refused to extend its *Benson* decision to computer programs generally, we find no warrant for our doing so.

Chatfield, 545 F.2d at 155 (citation omitted). The CCPA said it joined the Supreme Court in inviting "those who seek to preclude the patenting of all software or 'computer program' inventions [to] submit an appropriate proposal to the Congress." *Id.* at 156. Actually, the Supreme Court had invited Congressional action if Congress *wanted* programs to be patentable — the implication being that they might not be under current law — whereas the CCPA had determined that until Congress said no, it was not going to either.

Judge Rich's dissent in *Chatfield* (in which Judge Lane joined) raised a number of questions about the meaning of the Supreme Court's *Benson* decision. These questions, Judge Rich insisted, were in urgent need of being settled. *Id.* at 161. As in his *Johnston* dissent, Judge Rich asked the Supreme Court to review the case because the issues were obviously not going to be settled before the CCPA. *Id.*

[167] In *Chatfield*, the CCPA majority relied on the Supreme Court's definition of "algorithm" in *Benson* as limiting the meaning of the term to methods for solving a mathematical problem. The CCPA found no mathematical formula in Chatfield's main claims. Although some of his dependent claims did include formulae, the CCPA regarded the claims as ones not just "for" the formulae (that is, the claims did not preempt all uses of the formulae). For these reasons, the CCPA ruled in favor of the patentability of Chatfield's invention. *Chatfield*, 545 F.2d at 158-59.

Footnote 5 of the *Chatfield* opinion warns:

> Overconcentration on the word "algorithm" alone . . . may mislead. The Supreme Court carefully supplied a definition of the particular algorithm before it, i.e., "[a] procedure for solving a given type of mathematical problem." The broader definition of algorithm is "a step-by-step procedure for solving a problem or accomplishing some end." It is axiomatic that inventive minds seek and develop solutions to problems and step-by-step solutions often attain the status of patentable inventions. It would be unnecessarily detrimental to

Noll and *Chatfield* are also significant because after the Patent Office petitioned for certiorari on them, the CCPA united in its narrow construction of *Benson*. No further dissents based on a broader reading of *Benson* occurred thereafter,[168] even after yet another of the CCPA's rulings was overturned two years later in the Supreme Court's *Flook* decision.[169] Although the Supreme Court did not grant the Office's petitions in *Noll* or *Chatfield*,[170] the Patent Office was still not persuaded by the CCPA's interpretation of *Benson*, and the war between them continued.

> our patent system to deny inventors patent protection on the sole ground that their contribution could be broadly termed an "algorithm."

Id. at 156 n.5 (citation omitted). This footnote contains the seed of the distinction which blooms in later decisions between mathematical and nonmathematical algorithms. *See In re* Freeman, 573 F.2d 1237 (C.C.P.A. 1978); *see also infra* notes 178-88 and accompanying text.

The CCPA insisted in *Chatfield* that if one construed "algorithm" more broadly than "mathematical formula," all processes would be unpatentable. This, the CCPA said, would obviously be inconsistent with the Congressional decision to include "processes" as patentable subject matter. The CCPA therefore persuaded itself that it had to confine "algorithm" as it did.

The CCPA, by this discussion, set up a false distinction. A computer scientist's use of the term "algorithm" does have a broader meaning than "mathematical formula"; however, it has a less broad meaning than all processes in the world. An algorithm for a computer program represents the basic design for how data is to be organized and manipulated in a program in order to accomplish certain results. For a further discussion of mathematics and algorithms, see *infra* notes 389-94 and accompanying text.

[168] Curiously enough, the dissension within the CCPA ended before the Supreme Court denied the petition for certiorari in *Noll*, even though some other pro-patentability decisions came down after the *Noll* petition was filed. After *Noll* and *Chatfield*, the only CCPA case involving program-related claims in which there was a dissent that might have been based on a broader reading of *Benson* was *In re* Freeman, 573 F.2d 1237 (C.C.P.A. 1978) (Lane, J., dissenting without opinion).

[169] Parker v. Flook, 437 U.S. 584 (1978).

[170] It is easy to explain why the Court did not take the *Chatfield* case — because the Patent Office did not make a timely filing of the petition. It is more difficult to know what to make of the Court's denial of the petition for certiorari in the *Noll* case. *Noll* was only the third program-related patentability case on which the CCPA had ruled since the Supreme Court's *Benson* decision. Having already taken review of the only other ruling in that period in which the CCPA had found a program-related invention to be patentable and reversed it (albeit on other grounds), the Supreme Court may simply have wanted the law to develop a little more before taking another case. After all, the Supreme Court could not be in the business of reviewing every one of the CCPA's decisions in computer program cases.

However, the Court may also have been affected in its decision not to take the *Noll* case by the fact that the Noll invention was claimed in apparatus form and seemed to be an integrated part of a hardware system for displaying graphical images. (A television set is also a machine system for displaying graphical images.) Thus, the Court may not have been as distressed at the possibility of a patent in *Noll* as it had been in the *Benson* case.

C. *The Second Post-*Benson *Phase: Creating a Test to Limit* Benson's *Application*

In 1977 and 1978, between the time that the Patent Office petitioned for certiorari of the *Noll* and *Chatfield* decisions and the Supreme Court's ruling in *Flook*, the CCPA handed down seven additional rulings on the subject matter patentability of computer program-related inventions. In four cases, the CCPA overturned the Patent Office's rejections of the claims on the ground that the Office was construing *Benson* too broadly.[171] In the other three cases, the CCPA affirmed the Office's rulings,[172] although usually with some criticism for the rationale used by the Office to support its conclusion.[173]

On several occasions during this second phase, the CCPA changed its analysis of program-related claims, sometimes by reinterpreting some of its earlier decisions. During this second phase, for example, the CCPA

---

[171] *In re* Toma, 575 F.2d 872 (C.C.P.A. 1978) (automated natural language translation process, said not to involve an "algorithm in the *Benson* sense"); *In re* Freeman, 573 F.2d 1237 (C.C.P.A. 1978) (system for arranging mathematical equations for typesetting, also said not to involve an "algorithm in the *Benson* sense"); *In re* Flook, 559 F.2d 21 (C.C.P.A. 1977) (process for updating alarm limits for a catalytic conversion process, said to make use of an equation but not to be "for" it), *rev'd sub nom.* Parker v. Flook, 437 U.S. 584 (1978); *In re* Deutsch, 553 F.2d 689 (C.C.P.A. 1977) (process for optimizing the operation of multiple plant businesses, said to make only "incidental use" of computer programs and equations; "incidental use" could have, but did not, become yet another CCPA test for patentability of program-related inventions).

[172] *In re* Richman, 563 F.2d 1026 (C.C.P.A. 1977) (equation for a radar signal processing system); *In re* de Castelet, 562 F.2d 1236 (C.C.P.A. 1977) (equation for a machine tooling system); *Waldbaum II*, 559 F.2d 611 (C.C.P.A. 1977) (method of counting 0's and 1's in a data string; this decision, in effect, reversed a pre-*Benson* CCPA ruling on the same claims; *see supra* note 56).

[173] *See, e.g., In re* de Castelet, 562 F.2d 1236 (C.C.P.A. 1977). The *de Castelet* opinion criticized the Patent Office for interpreting *Benson* "as foreclosing patentability of the appealed claims" because:

> [A]s de Castelet admitted at oral argument, the only practical application of the involved algorithm (equations) is in the claimed method. But the notion that the "nutshell" language makes a method nonstatutory whenever it involves mathematical equations having their only practical use in the method is neither impelled by the rationale of *Benson* nor supported by precedent.

Id. at 1241; *see also id.* at 1240 (rejecting the Office's argument that the "thrust" of *Benson* was that computer programs or algorithms were not patentable until or unless Congress said so).

Criticism of the Patent Office's approach to program-related claims was, however, sharpest in cases in which the CCPA reversed the Office's ruling. *See, e.g., In re* Freeman, 573 F.2d 1237, 1243 n.2 (C.C.P.A. 1978) (criticizing the Patent Office for "mudd[ying]" the appeal with "disingenuous presentations" which "disserves the need for clarity in the law, and unjustly skews the judicial process").

explicitly abandoned the most specious of its earlier interpretations of *Benson* as only applying to claims drafted in "method" form.[174] In addition, the CCPA was sometimes willing to say that *Benson*'s proscriptions could not be avoided merely by field of use limitations in the claims.[175] In one case, the CCPA even expanded its interpretation of *Benson* as prohibiting not only claims for equations but also claims for methods of calculation.[176]

Occasionally, the CCPA would, perhaps inadvertently, make a statement that would call into question the patentability of computer program innovations.[177] The most important developments in this second post-

---

[174] *Freeman*, 573 F.2d at 1247. Although the CCPA, after *Noll*, had stopped referring to *Benson* as applying only to method claims, it was not until *Freeman*, the sixth decision in this second phase, that the CCPA repudiated its earlier view that *Benson* did not apply to claims drafted in apparatus form. *Freeman* was handed down while the Supreme Court was considering whether to reverse the CCPA's *Flook* decision.

[175] *See Waldbaum II*, 559 F.2d 611, 617 (C.C.P.A. 1977) (rejecting the argument that because Waldbaum had limited the scope of his claims — some to data processing applications, and some to telephone service applications — there were no *Benson* problems with the claims; a patent on these claims "would, in practical effect, be a patent on the algorithm itself — albeit in its limited, specific application to *calculating* the number of busy and idle lines in a telephone system."); *see also In re* Richman, 563 F.2d 1026 (C.C.P.A. 1977) (rejecting the argument that limitation to a particular technological field could be used to avoid *Benson's* proscriptions). To the extent that *Noll* and *Chatfield* seem to endorse the view that end use or technological environment limitations could save claims from being found unpatentable under *Benson*, *see supra* note 165, *Waldbaum II* seems to suggest that the CCPA cases which endorse such limitations were also "false starts" by the CCPA. *But see Freeman*, 573 F.2d at 1245 (this test made unpatentable only claims that would preempt all uses of a mathematical algorithm); *In re* Deutsch, 553 F.2d 689, 692 (C.C.P.A. 1977) (suggesting "end use" limitations could save claims under *Benson*).

[176] *Waldbaum II*, 559 F.2d 611, was the case in which the CCPA interpreted *Benson* as applying to "methods of calculation" as well as to equations. Waldbaum tried to argue that *Benson* did not apply because his claims did not include an equation, relying on *Noll* and *Chatfield*, in which the CCPA seemed to have limited *Benson*'s reach only to cases involving equations. However, in *Waldbaum II*, the CCPA agreed with the Patent Office's characterization of Waldbaum's process as one for a "method of calculation," to which *Benson* did in fact apply. *Id.* at 616-17.

To the extent that *Chatfield* suggests that *Benson* was only concerned with "mathematical formulae," *Waldbaum II* also suggests that this aspect of *Chatfield* was a "false start" by the CCPA. Nevertheless, the predominance of equations in the analysis of subsequent cases makes this less clear than the abandonment of other "false starts."

It is worth emphasizing the CCPA's "false starts" because it has not always been recognized that the shifting sands of the CCPA's decisions and reinterpretations of its prior decisions have contributed to the uncertainty of the law surrounding computer program patents.

[177] The CCPA characterized the process claimed in *de Castelet* as follows:

> Overall, therefore, de Castelet's claimed method involves the storing of certain mathematical data in a computer, inputting additional mathematical data, causing the computer to perform programmed computations using the stored and inputted data, and finally causing

Benson phase, however, were the development of a "test" for patentability of program-related claims and the struggle over use of the "point of novelty" test, both of which are discussed in their own subsections below.

1. The Two-Step Freeman *Test*

The most important doctrinal development during this period was the creation of a two-part "test" for judging whether *Benson* forbade issuance of a patent on the claims, which the CCPA announced in *In re Freeman*[178] while the *Flook* case was under review by the Supreme Court.[179] Unlike many of the other approaches that the CCPA had employed in judging program-related claims, the *Freeman* test was one that the CCPA used for many years. This test, however, may have been used longer than it should have been, given the implicit criticism of it by the Supreme Court in *Flook*.[180] The *Freeman* test was derived from the Supreme

---

the computer to transmit the results of those computations to a "model forming means." *De Castelet*, 562 F.2d at 1244.

This characterization of de Castelet's invention would seem to raise questions about the patentability of all programs. All computer programs do essentially the same thing as de Castelet's invention. It is curious that the CCPA here discusses programs as if it still had not grasped what computer program-related patent cases were really about.

Even though de Castelet tried to argue that his claims were not just for an equation, the CCPA said it was convinced "that the resolution of de Castelet's equations by a computer, which then transmits the electrical result to a tool, is not a practical application within the rule, and that a patent containing the appealed claims would in effect be no more than a patent on de Castelet's equations." *Id.* at 1245. Although the CCPA said nothing critical about the *Bernhart* decision in *de Castelet*, it would appear that had the CCPA in *Bernhart* followed its reasoning in *de Castelet*, the court would have rejected the Bernhart claims.

[178] 573 F.2d 1237, 1245 (C.C.P.A. 1978). Freeman's invention was said to be an improved method and apparatus for accurate positioning of mathematical symbols for typesetting purposes. His system analyzed the positions of mathematical symbols in mathematical formulae (or other mathematical expressions) as if they were on a north/south/east/west axis at certain "concatenation points." The system would then build a hierarchical tree structure for storing data on those positions. Using a local positioning algorithm, the hierarchical tree structure, and the concatenation points, Freeman's programmed system was said to be able to typeset the mathematical formulae with greater accuracy than previous methods had done. *Id.* at 1239-40.

[179] The *Freeman* opinion was issued by the CCPA on March 30, 1978. The *Flook* case was argued before the Supreme Court on April 25, 1978. The Supreme Court's *Flook* decision was issued on June 22, 1978. As indicated *supra* note 174 and accompanying text, it was not until *Freeman* that the CCPA explicitly rejected the "apparatus" versus "method" distinction it had used in the first post-*Benson* phase.

[180] *See infra* note 210 and accompanying text.

Court's "nutshell" in *Benson*.[181] The first step of the *Freeman* test was to determine whether the claim recited a "mathematical algorithm."[182] The second step was to inquire whether a patent on this "mathematical algorithm" would "wholly pre-empt[]" use of it.[183] On its face, the *Freeman* test seemed to revert to the *Noll* and *Chatfield* position that field of use limitations were enough to save mathematical claims from *Benson*'s proscriptions,[184] and indeed the CCPA's *Flook* decision suggests that this reversion had taken place.[185]

Adoption of the *Freeman* test had some important implications. First, if the claims contained no reference to equations, the CCPA would generally find that *Benson* simply did not apply.[186] A second implication was that

---

[181] Although the *Benson* "nutshell" had been used in a number of earlier cases, *see, e.g., In re* Deutsch, 553 F.2d 689 (C.C.P.A. 1977), it was not clearly articulated as a "test" for judging patentability of program-related inventions until *Freeman*. For more discussion of the later evolution of this test, see *infra* notes 226-32 and accompanying text.

[182] *Freeman*, 573 F.2d at 1245. As in *Chatfield*, *supra* note 167, the CCPA in *Freeman* warned against an overbroad interpretation of "algorithm." *Id.* "Because every process may be characterized as a 'step by step' procedure . . . for accomplishing some end, a refusal to recognize that *Benson* was only concerned with *mathematical* algorithms leads to the absurd view that the Court was reading the word 'process' out of the statute." *Id.* at 1246 (emphasis in original). The CCPA in *Freeman* did insist that a claim substituting words for a mathematical formula that meant the same thing would be judged as a claim that recited a formula directly. The CCPA also pointed out that *Benson* itself was a case in which prose equivalent to mathematical expressions had been recited in the claims. *Id.*

[183] *Id.* at 1245.

[184] In *Waldbaum II* and *Richman*, both decided in 1977, *see supra* note 175, the CCPA seemed to hold that field of use limitations were not enough to satisfy *Benson*. Now in *Freeman*, the CCPA seemed to revert to the 1976 approach in *Noll* and *Chatfield* of analyzing *Benson*. *See supra* note 165.

[185] For a discussion of the CCPA's *Flook* decision and the Supreme Court's response to it, see *infra* notes 197-214 and accompanying text. The Supreme Court seems to have perceived *Flook* as a "field of use" limitation case. *See infra* note 207.

[186] *See, e.g., In re* Toma, 575 F.2d 872 (C.C.P.A. 1978); *In re* Freeman, 573 F.2d 1237 (C.C.P.A. 1978) (both dispensing with challenges to the patentability of program-related inventions by finding that since no " 'algorithm' in the *Benson* sense" was being claimed, no patentability problem existed. *Freeman*, 573 F.2d at 1245.). *Toma* presented some particularly interesting patentable subject matter questions, since Toma's claim was for an algorithm for a computerized natural language translation process. The Patent Office rejected the claim because it was for an algorithm under *Benson* and for a mental process, and because it made a contribution to the liberal, not the technological, arts. Toma's claim, however, was ruled patentable under the *Freeman* test because the claim did not recite an equation. For a lengthy discussion of *Toma*, see *infra* notes 402-05 and accompanying text.

With the exception of *Waldbaum II*, 559 F.2d 611 (C.C.P.A. 1977), said to involve a method of counting, *see supra* note 175, every case in the period between *Benson* and *Diehr* in which the CCPA affirmed the rejection of program-related claims involved claims for equations. It was only after *Diehr*

the *Freeman* test allowed the CCPA to make a clear distinction between mathematical algorithms and other algorithms. The CCPA used this distinction as a basis for criticizing the Patent Office for taking too broad a view of " 'algorithm[s]' in the *Benson* sense."[187] The problem, the CCPA once again insisted, was not with patenting computer programs but only with patenting mathematics.[188]

2. *The "Point of Novelty" Dispute and the* Flook *Case*

In addition to their continuing battle over what the Supreme Court had meant in *Benson* by the term "algorithm," the Patent Office and the CCPA feuded vigorously during this second post-*Benson* period over the Patent Office's use of a "point of novelty" test as a way of judging whether an applicant with a program-related invention was claiming statutory subject matter.[189] In a substantial number of cases, the Patent Office

---

that the CCPA (and its successor, the CAFC) seemed to broaden its view of the meaning of "algorithm." *See infra* notes 367-76 and accompanying text.

[187] *In re* Freeman, 573 F.2d 1237, 1245 (C.C.P.A. 1978). The CCPA described the Patent Office's opinion to be that "every implementation with a programmed computer equals 'algorithm' in the *Benson* sense." *Id.* The CCPA regarded the "local positioning algorithm" in the Freeman application as being an algorithm "in its broad sense, i.e., to identify a step-by-step procedure for accomplishing a given result," rather than being a procedure for solving a mathematical problem. *Id.* at 1246.

The *Freeman* case is one of a number of CCPA cases that misconstrues the nature of the invention being claimed. Freeman's invention did involve a mathematical problem, not because the items to be typeset were mathematical equations, but because the concatenation points, the hierarchical tree for noting positions, and the positioning algorithm were mathematical in character. However, the only thing the CCPA focused on in the Freeman claim was a typesetting process.

See also In re Toma, 575 F.2d 872, 877 (C.C.P.A. 1978) ("Translating between natural languages is not a mathematical problem as we understand the term to have been used in *Benson*. Nor are any of the recited steps in the claims mere procedures for solving mathematical problems."). *But see infra* notes 390-94 and accompanying text (explaining why Toma's claims, properly understood, are mathematical in character).

[188] *See, e.g., In re* Freeman, 573 F.2d 1237, 1244-45 (C.C.P.A. 1978) ("That computer programs are not patentable was neither the holding nor the 'thrust' of *Benson*."). "Mathematical" procedures were all the Court had been concerned with in *Benson*. *See also In re* Deutsch, 553 F.2d 689, 693 (C.C.P.A. 1977) ("Though ably argued by the Solicitor, the questions of whether the programs fed to the computers described in Deutsch's specification are 'processing' programs, and whether programs are themselves patentable, are . . . not relevant on this appeal.").

[189] There is an interesting parallel between the post-*Benson* "point of novelty" debate and the pre-*Benson* "mental process" debate. As in the pre-*Benson* debate, the Patent Office, relying on an earlier CCPA decision, took an analytic position that tends to limit the extent of patent protection available to computer program-related inventions. As in the pre-*Benson* debate, the CCPA decided to

rejected applications involving computer program-related inventions on the ground that the only "point of novelty" between a particular applicant's claims and the state of the prior art was an algorithm.[190] The CCPA, although it had seemingly endorsed this approach in its first post-*Benson* decision, *In re Christensen*,[191] spent most of this second post-*Benson* phase renouncing this approach in arguing that subject matter determinations should be made by looking at the subject matter claimed as a whole, rather than by dissecting the claims into new and old elements.[192]

---

repudiate one of its earlier rulings and adopt a different approach, one that tends to increase the availability of patent protection for program-related inventions. Despite clear disapproval by the CCPA, the Patent Office persisted doing as it thought right. The CCPA, expressing impatience with the Office for not complying with its directives, would then overturn the Office's rulings. When the Supreme Court finally took one of the cases reflecting this battle for review (first *Benson*, then *Flook*), the Court seemed to vindicate the Patent Office's stubbornness (although, as becomes apparent, the victories were relatively short-lived).

[190] Although the discussion is concentrated on *In re* Flook, 559 F.2d 21 (C.C.P.A. 1977), *rev'd sub nom.* Parker v. Flook, 437 U.S. 584 (1978) (discussed *infra* notes 197-213 and accompanying text), there are numerous other "point of novelty" cases in this second phase. *E.g.*, *In re* Toma, 575 F.2d 872 (C.C.P.A. 1978); *In re* Freeman, 573 F.2d 1237 (C.C.P.A. 1978); *In re* de Castelet, 562 F.2d 1236 (C.C.P.A. 1977).

[191] 478 F.2d 1392, 1394 (C.C.P.A. 1973), *overruled*, *In re* Taner, 681 F.2d 787 (C.C.P.A. 1982) (concluding that "a method claim in which the point of novelty is a mathematical equation to be solved as the final step of the method [is not] a statutory method. . . . Given that the method of solving a mathematical equation may not be the subject of patent protection, it follows that the addition of old and necessary antecedent steps of establishing values for the variables in the equation cannot convert the unpatentable method to patentable subject matter."); *see supra* notes 151-53 and accompanying text.

Christensen was not the first CCPA case to use a "point of novelty" approach for testing patentability. Earlier, the CCPA seemed to endorse this test in *In re* Abrams, 188 F.2d 165 (C.C.P.A. 1951). For a discussion of *Abrams*, see *supra* notes 30-35 and accompanying text. The Patent Office also used this approach in analyzing program-related claims in the pre-*Benson* era. *See, e.g.*, *In re* Musgrave, 431 F.2d 882 (C.C.P.A. 1970) (only new thing was equations); *In re* Bernhart, 417 F.2d 1395 (C.C.P.A. 1969) (same). In both *Bernhart* and *Musgrave*, the CCPA criticized use of the "point of novelty" approach although it briefly moved back to an endorsement of the test in *Christensen*.

[192] Thus, it did not matter to the CCPA if the only element in the claimed process that was not already in the state of the art was an algorithm, so long as more than the algorithm was recited in the claim. *See, e.g.*, *In re* Flook, 559 F.2d 21 (C.C.P.A. 1977), *rev'd sub nom.* Parker v. Flook, 437 U.S. 584 (1978); *see also In re* Sherwood, 613 F.2d 809 (C.C.P.A. 1980).

Although a "point of novelty" approach to judging the patentability of program-related inventions has, in the aftermath of Diamond v. Diehr, 450 U.S. 175 (1981), fallen out of favor in the United States, it is worth noting that this standard is widely used abroad. *See, e.g.*, H. HANNEMAN, *supra* note 37, at 125, 226-27.

The Court of Appeals for the Federal Circuit has used a "point of novelty" test in design patent cases. However, it does so not in making subject matter determinations, but in judging the scope of design patent claims in deciding whether infringement has occurred. Use of the test confines the reach of the design patent to that which distinguishes the invention from the prior art. *See, e.g.*, Litton Sys.,

The *Flook* case is representative of this second debate since it was decided on a "point of novelty" basis by the Patent Office and reversed by the CCPA because of use of this test. The CCPA's ruling in *Flook* was subsequently reversed by the Supreme Court on a basis that seemed, on the face of it, to endorse the Patent Office's approach, even though the Court never explicitly used "point of novelty" terminology.[193]

Flook's claims related to an improved process for updating alarm limits for plants that perform catalytic conversions of petrochemicals.[194] The patent examiner acknowledged that Flook's invention was useful and within the technological arts, but denied the patent because the only respect in which his process differed from the prior art was its use of an equation in the second step of the process.[195] Because of this, the examiner thought that to grant a patent on Flook's claims would be to grant a patent on the equation itself, which *Benson* forbade. The Patent Board of Appeals agreed.[196]

The CCPA posed the issue in *Flook* quite differently than the Patent Office had. It thought that the issue was "whether a claim to a process which *uses* an algorithm to modify a conventional manufacturing system is statutory subject matter."[197] The CCPA denied that it had ever en-

---

Inc. v. Whirlpool Corp., 728 F.2d 1423 (Fed. Cir. 1984).

[193] *In re* Flook, 559 F.2d 21 (C.C.P.A. 1977), *rev'd sub nom.* Parker v. Flook, 437 U.S. 584 (1978). To see the apparent Supreme Court endorsement of a "point of novelty" approach, see *infra* notes 201, 206-08, and accompanying texts.

[194] Catalytic conversion involves breaking down oil products into constituent chemicals by subjecting them to certain heat and pressure conditions. Alarm limits, which represent the safest heat and pressure conditions for catalytic conversion, are used to signal the onset of dangerous conditions that require correction. As the plant goes through different phases of its operation, alarm limits must be "updated" to reflect different states of operation of the plant. *Flook*, 437 U.S. at 585.

[195] *Flook*, 559 F.2d at 22. The first step of the Flook process involves measuring the relevant variables involved in the chemical process; the second step puts the data into the equation; the third step updates the alarm limits for the plant in accordance with the results of the equation. To see how the CCPA later interprets the Flook process, see *infra* note 219.

[196] *Flook*, 559 F.2d at 22.

[197] *Id.* (emphasis in original). Note the emphasis here on "conventional manufacturing system" in the CCPA's statement of the issue in the *Flook* case. *See also In re* Deutsch, 553 F.2d 689 (C.C.P.A. 1977) (characterizing claims for a method of optimizing multiplant operations as if they were for conventional industrial processes when they appear to have been methods of optimizing business planning for the plants since they involved price and cost calculations). As shall be seen, characterizations of processes as "industrial" become more common in the third post-*Benson* phase. *See infra* notes 240-47 and accompanying text. The Supreme Court was not impressed by the "conventional manufacturing system" language from the CCPA's *Flook* opinion, viewing it as a kind of

dorsed a "point of novelty" approach in *Christensen* and asserted that, in any event, it was an inappropriate approach to analysis of such claims.[198] All that *Benson* required, said the CCPA, was to claim something that "materially limits the claim to a scope less than the mere act of solving an algorithm."[199] The CCPA indicated that this requirement could "not [be] satisfied by the recitation of data gathering steps," but it might be "satisfied by the recitation of some sort of post-solution activity."[200] Adjustment of the alarm limit in *Flook* was enough post-solution activity to satisfy the CCPA.

---

field of use limitation which was insufficient to save claims from Benson's proscriptions. *See infra* notes 207 and 210.

[198] For a discussion of the CCPA's rationale for rejecting the Christensen claims, see *supra* notes 151-53 and accompanying text. *Christensen* was reinterpreted in *In re* Chatfield, 545 F.2d 152, 158 (C.C.P.A. 1976), as a case in which the claims were unpatentable because the claimed process involved only data gathering and equation-solving steps, rather than because of where its "point of novelty" did or did not lie.

Flook argued for a different interpretation of *Christensen*. In his view, the process in *Christensen* was unpatentable only because the "last step" in that process was solving an equation. Since the last step in Flook's process was the updating of alarm limits, not solving an equation, he argued his claims should not be rejected. *Flook*, 559 F.2d at 22. The CCPA agreed with Flook that the holding in *Christensen* was "expressly limited" to claims in which "nothing is done after solution of the algorithm." *Id.* at 23.

Only a short time thereafter, however, in *In re* Richman, 563 F.2d 1026 (C.C.P.A. 1977), the CCPA once again reinterpreted *Christensen*, saying that neither the fact that an equation is the last step in a claimed process nor the order in which the steps are set forth in the claims is determinative of the patentability issue. This decision seems to have reinstated *Chatfield*'s interpretation of *Christensen*, which made the "last step" test for patentability one more in the long line of CCPA "false starts" in computer program cases.

Professor Chisum has argued that the CCPA had, in effect, overruled *Christensen* on this point in *In re* Taner, 681 F.2d 787 (C.C.P.A. 1982). Chisum, *supra* note 12, at 1002. However, the Court of Appeals for the Federal Circuit recently made clear that this is not so. *See In re* Grams, 888 F.2d 835, 839 (Fed. Cir. 1989). For further discussion of the continuing vitality of the *Chatfield* interpretation of *Christensen*, see *infra* notes 361-77 and accompanying text.

[199] *Flook*, 559 F.2d at 23. In stating that *Benson* only required some kind of limitation on the scope of a claim such that there would not be infringement merely by solving the equation, the CCPA opinion in *Flook* seemed to revive notions which appeared in earlier cases, such as the notion that "end use" limitations might be enough to satisfy *Benson*. *See, e.g., In re* Noll, 545 F.2d 141 (C.C.P.A. 1976); *supra* note 165 and accompanying text. It is worth noting that only a week before its ruling in *Flook*, the CCPA ruled in *Waldbaum II*, 559 F.2d 611 (C.C.P.A. 1977), that field of use limitations were insufficient to save claims otherwise unpatentable under *Benson*. *See supra* note 175.

[200] *Flook*, 559 F.2d at 23. Because merely using the algorithm to perform calculations would not infringe the patent, the CCPA thought it did not "wholly pre-empt" use of the algorithm, and thus was safe under *Benson*. *Id.* In view of the Supreme Court's repudiation of the CCPA's ruling in *Flook*, the "post-solution activity limitation" test seems to represent yet another "false start" by the CCPA.

The Supreme Court posed the issue in *Flook* quite differently from the CCPA: "The question in this case is whether the identification of a limited category of useful, though conventional, post-solution applications of [a mathematical] formula makes respondent's method eligible for patent protection."[201] The Court ruled that it did not.[202]

As in *Benson*, the Court in *Flook* regarded the applicant's new mathematical procedure as unpatentable in the same way that a newly discovered scientific principle or law of nature would be.[203] Flook's argument — that *any* post-solution activity, no matter how conventional or obvious, could transform an unpatentable mathematical formula into a patentable process — was rejected by the Court as exalting form over substance. The Court perceived it as equivalent to saying that the discoverer of the Pythagorean theorem could patent it simply by including in the patent claim "a final step indicating that the formula, when solved, could be usefully applied to existing surveying techniques."[204] The Court considered this result absurd.

The Court insisted that it did not regard a process as outside the realm of patentability merely because it made use of a mathematical formula or law of nature.[205] But, said the Court, the "process itself, not merely the mathematical algorithm, must be new and useful."[206] In response to

---

[201] *Flook*, 437 U.S. at 585. Notice that the Court did not make use of "point of novelty" language in describing the issue of the case, yet its statement of the issue certainly seems consistent with the "point of novelty" approach.

[202] Justice Stewart wrote a dissenting opinion in the *Flook* case in which Justices Burger and Rehnquist joined. *See* Parker v. Flook, 437 U.S. 584, 598-600 (1978). The dissent, while agreeing with the majority that patents could not issue for processes, such as multiplication, thought that Flook's process was different because only one of its steps involved calculations. *Id.* at 598.

[203] *Id.* at 589.

[204] *Id.* at 590. For a discussion of this same issue in connection with the *Benson* and *Diehr* opinions and the philosophical issue that lies at the heart of the question of whether an equation is a patentable invention or an unpatentable discovery or scientific principle, see *supra* note 99, *infra* notes 272-74, and accompanying texts.

[205] *Flook*, 437 U.S. at 590. In response to Flook's argument that his process was patentable because his claims literally recited a process, the Court observed that its holding in *Benson* "foreclose[d] a purely literal reading of § 101." *Id.* at 589. The Court went on to say that the "concept of patentable subject matter under § 101 is not 'like a nose of wax which may be turned and twisted in any direction.'" *Id.* at 590.

[206] *Id.* at 591. The Court said:

> [T]he novelty of the mathematical algorithm is not a determining factor at all. Whether the algorithm was in fact known or unknown at the time of the claimed invention, as one of the "basic tools of science and technological work," it is treated as though it were a famil-

Flook's argument (which was also the CCPA's) that this approach improperly imported novelty (section 102) and nonobviousness (section 103) concerns into the subject matter determination, the Court repeated its concern that without requiring some novelty in the process apart from the algorithm, "the determination of patentable subject matter [would] depend simply on the draftsman's art [which] would ill serve the principles underlying the prohibition against patents for 'ideas' or phenomena of nature."[207] The Court explained:

> The rule that the discovery of a law of nature cannot be patented rests, not on the notion that natural phenomena are not processes, but rather on the more fundamental understanding that they are not the kind of "discoveries" that the statute was enacted to protect. The obligation to determine what type of discovery is sought to be patented must precede the determination of whether that discovery is, in fact, new or obvious.[208]

As in *Benson*, the Supreme Court in *Flook* seemed to be trying to steer a middle course between the positions of the Patent Office and the CCPA. On the one hand, the Court vindicated the Patent Office's analysis of the Flook claim and repudiated the CCPA's analysis. It reaffirmed its earlier ruling that algorithms were unpatentable,[209] thereby implicitly criticizing

---

iar part of the prior art.

Id. at 591-92 (citation omitted). Although not explicitly mentioning the "point of novelty" test, this statement by the Court of what was required in order for a process that included an algorithm to be patentable is certainly consistent with the "point of novelty" approach. Treating the unpatentable law of nature, principle, etc., as already in the state of the art has a long history in patent law. *See, e.g.*, O'Reilly v. Morse, 55 U.S. (15 How.) 402 (1853).

[207] *Flook*, 437 U.S. at 593. The Court insisted that its approach was consistent with looking at the subject matter being claimed as a whole. Flook's process, said the Court, was unpatentable under section 101, "not because it contains a mathematical algorithm as one component, but because once that algorithm is assumed to be within the prior art, the application, considered as a whole, contains no patentable invention." *Id.* at 594. The Court gave examples of the many things pertaining to Flook's process that were already in the prior art: the process of breaking down oil products by catalytic conversion, the practice of monitoring process variables, the calculation and use of alarm limits, the notion that alarm limits needed to be updated, and the use of computers to do automatic monitoring and calculations. *Id.*

All that *Flook* added to this was "a new and presumably better method for calculating alarm limit values." *Id.* at 594-95. The Court characterized its holding in *Flook* as ruling that "a claim for an improved method of calculation, even when tied to a specific end use, is unpatentable subject matter under section 101." *Id.* at 595 n.18.

[208] *Id.* at 593.

[209] *Id.* at 588-89.

the CCPA for taking too narrow a view of the *Benson* decision.[210] Furthermore, as in *Benson*, the Court seemed to suggest that Congress should take action on computer program patent issues.[211]

On the other hand, the *Flook* Court seemed to accept the CCPA interpretation of "algorithm" as equivalent to mathematical formulae and methods of calculation.[212] The Court also seemed to agree with the CCPA that *Benson* should not be construed as a blanket prohibition on computer program patents.[213] In addition, the Court insisted that its analysis of Flook's claim was not inconsistent with the CCPA's approach of viewing claims as a whole. The CCPA took this as approval of its rejection of the "point of novelty" test.[214]

Given the length of time that the Patent Office and the CCPA had been feuding about the correctness of their radically different approaches to judging the patentability of program-related inventions, it is not surpris-

---

[210] The Court's conclusion in *Flook* that the claimed process was unpatentable was a critique of the CCPA's analysis of the claims. The Court's proposed method of claim analysis in *Flook*-like cases (*i.e.*, as assuming the algorithm was already known and asking if there was anything inventive in the process) was also a critique of the CCPA's approach. Most directly, the Court rejected the CCPA's views in concluding that it was not enough to save Flook's claim that a patent for the process would not wholly preempt use of the equation, and that it was not sufficient that the claims might involve some post-solution activity or be limited to a particular field of use, such as in catalytic conversion. *See id.* at 589-90, 593, 595 n.18.

[211] Although troubled by assertions made by the government's attorneys that the CCPA's approach to computer program patents would have a debilitating effect on the computer software industry and that a clear statement of the extent of patent protection for computer programs was needed, the Court in *Flook* once again demurred, saying that there were "[d]ifficult questions of policy concerning the kinds of programs that may be appropriate for patent protection and the form and duration of such protection [that] can be answered by Congress on the basis of current empirical data not equally available to this tribunal." *Id.* at 595. The Court reasoned that its duty lay in construing the patent statute in light of the precedents, and to "proceed cautiously when we are asked to extend patent rights into areas wholly unforeseen by Congress." *Id.* at 596.

[212] A footnote to the Court's description of the second step of Flook's process indicates that the Court accepted the CCPA's interpretation of algorithm as referring to equations. *Id.* at 585 n.1.

[213] In a footnote, the *Flook* Court implicitly criticized the Patent Office for reading its *Benson* decision too broadly by adverting to an "argument" (without saying whose) that programs are not patentable because "process" had been confined in its meaning (by judicial interpretation) to processes that transform matter. The Court responded to this "argument" by saying "[a]s in *Benson*, we assume that a valid process patent may issue even if it does not meet one of these qualifications of our earlier precedents." *Id.* at 588 n.9. This statement, like its counterpart in *Benson*, indicates the Court's desire not to close the door entirely to computer program patents, but it is far from a hearty endorsement of the CCPA's position.

[214] *See infra* note 219 and accompanying text.

ing that neither combatant saw the middle course that the Court may have been trying to steer in *Flook*. Instead, each side seemed to have gotten mixed signals about who was "winning." Because of this confusion, the *Flook* decision, like *Benson* before it, did not end the battle between these protagonists, but only set the stage for yet another round of skirmishes.[215]

D. *From* Flook *to* Diehr*: A New Emphasis on the Industrial Character of Program-Related Inventions*

There were nine CCPA decisions on the patentability of computer program-related inventions during the three years between the Supreme Court decisions in *Flook* and *Diehr*. The CCPA overturned the Patent Office's subject matter rejections of these claims in five of these cases[216] and upheld them in four others.[217]

In this third post-*Benson* phase, the Patent Office, emboldened by its third Supreme Court victory in *Flook*, seemed to become more insistent than before on its broad interpretation of *Benson*.[218] The CCPA, however, was equally insistent on its narrow interpretation of the case and

---

[215] *See, e.g.*, Comment, *Computer Program Patentability — The CCPA Refuses to Follow the Lead of the Supreme Court in* Parker v. Flook, 58 N.C.L. REV. 319 (1980) (noting how little the *Flook* decision changed the CCPA's analysis of computer program-related claims).

[216] *In re* Sherwood, 613 F.2d 809 (C.C.P.A. 1980) (improved method of producing accurate seismic map), *cert. denied*, 450 U.S. 994 (1981); *In re* Phillips, 608 F.2d 879 (C.C.P.A. 1979) (claims for computerized architectural plan drafting); *In re* Diehr, 602 F.2d 982 (C.C.P.A. 1979), *aff'd sub nom.* Diamond v. Diehr, 450 U.S. 175 (1981) (claims for an improved rubber curing process which included a computer program as an element); *In re* Bradley, 600 F.2d 807 (C.C.P.A. 1979), *aff'd sub nom.* Diamond v. Bradley, 450 U.S. 381 (1981) (claims for improved method of structuring data for microcode function); *In re* Johnson, 589 F.2d 1070 (C.C.P.A. 1978) (claims for methods of removing "noise" from seismic data).

[217] *In re* Walter, 618 F.2d 758 (C.C.P.A. 1980) (claims for improved method of seismic analysis); *In re* Maucorps, 609 F.2d 481 (C.C.P.A. 1979) (apparatus claims for system of optimizing sales organizations); *In re* Gelnovatch, 595 F.2d 32 (C.C.P.A. 1979) (claims for method of designing microwave circuits); *In re* Sarkar, 588 F.2d 1330 (C.C.P.A. 1978) (claims for method of analyzing water flow).

[218] In the first several CCPA cases decided between the time of the Supreme Court decisions in *Flook* and *Diehr*, the Patent Office rejections of program-related claims seemed to become broader in scope and less detailed in analysis than they had been in the second post-*Benson* phase. This caused the CCPA some irritation and finally provoked a remand for a more detailed analysis of the claims and rationale for rejecting them. *See, e.g.*, *In re* Phillips, 593 F.2d 1021 (C.C.P.A. 1979) (remanded for a supplemental opinion; insufficient to say that the claims were for a program that had its only practical application in a digital computer and were therefore unpatentable under *Benson*).

saw *Benson* as endorsing this interpretation.[219] The CCPA found answers to every new argument that the Patent Office devised about why claims for program-related inventions should be denied[220] and warned of the

---

[219] *See, e.g., In re* Johnson, 589 F.2d 1070, 1075-78 (C.C.P.A. 1978). The CCPA in *Johnson* went to great lengths to explain why it thought that the Supreme Court in *Flook* had endorsed the CCPA's analysis of program-related claims, saying, "While a degree of uncertainty existed concerning the proper interpretation of *Gottschalk v. Benson* . . . , it is clear after *Flook* that the board's conclusion that patent protection is proscribed for all inventions 'algorithmic in character' is overbroad and erroneous." *Id.* at 1075. Despite the fact that the Court in *Flook* upheld the Patent Office's rejection of Flook's application because the only novel thing about his process was an algorithm, *see supra* notes 201-02 and accompanying text, the CCPA did not see in this an endorsement of the Patent Office's "point of novelty" approach. Rather, the CCPA read the Court in *Flook* to have endorsed its "subject matter as a whole" approach. *Id.* at 1077 n.7 (citing 35 U.S.C. § 103 (1984)). In reading the *Johnson* opinion, one might get the impression that the Supreme Court had affirmed the CCPA's ruling in *Flook* rather than reversing it and criticizing its analysis.

However, in *Johnson*, the CCPA finally offered some explanation of why the Supreme Court ruled the claims to be unpatentable in *Flook*. 589 F.2d at 1076, 1078. It did so indirectly by offering two distinctions between Johnson's claims and those of Flook. Flook's claims, the CCPA first observed, recited a novel mathematical formula, whereas Johnson's claims did not allege a new mathematical procedure (was Flook's mistake in admitting that his equation was the *only* new element in his process?). Second, the CCPA observed that the purpose of performing Flook's process was to compute a number, whereas the purpose of Johnson's process was to produce a more accurate seismic record. *Id.* at 1078. (For discussion of printed output as a distinguishing feature in cases involving program-related claims, see *infra* notes 233-39 and accompanying text.) *But see* Comment, *supra* note 215, at 347-48 (pointing to inconsistencies between how the CCPA previously characterized the Flook process and how it characterized it in *Johnson*, contrary to what the Supreme Court intended in *Flook*). In a later case, the CCPA characterized Flook's process as unpatentable because the result of that process had been a number (*i.e.*, the updated alarm limit). *See In re* Walter, 618 F.2d 758, 768 (C.C.P.A. 1980).

[220] *See, e.g., In re* Bradley, 600 F.2d 807 (C.C.P.A. 1979) (rejecting the Patent Office's argument that because digital computers operate through use of a numerical radix, methods of operating them were inherently mathematical in character and, hence, unpatentable under *Benson* and *Flook*), *aff'd by an equally divided court sub nom.* Diamond v. Bradley, 450 U.S. 381 (1981). Noting that adoption of this characterization would make all computer program-related inventions unpatentable, the CCPA insisted that this argument confused *what* a computer does with *how* it does it:

> It is of course true that a modern digital computer manipulates data, usually in binary form, by performing mathematical operations, such as addition, subtraction, multiplication, division, or bit shifting, on the data. But this is only *how* the computer does what it does. Of importance is the significance of the data and their manipulation in the real world, i.e., *what* the computer is doing. It may represent the solution of the Pythagorean theorem, or a complex vector equation describing the behavior of a rocket in flight, in which case the computer is performing a mathematical algorithm and solving an equation. This is what was involved in *Benson* and *Flook*. On the other hand, it may be that *the data and the manipulations performed thereon* by the computer, when viewed on the human level, *represent the contents of a page of the Milwaukee telephone directory, or the text of a court opinion* retrieved by a computerized law service. *Such information is utterly devoid of mathematical significance.* Thus the board's analysis does nothing but provide a quick and

harm that would come to this new field from denying patent protection by an overbroad interpretation of *Benson* and *Flook*.[221] The CCPA was quite insistent that unless a case fit within an already established exception to patentable subject matter, such as the exception for "mathematical algorithms," it was inappropriate to raise subject matter questions.[222]

As compared with the second post-*Benson* period, there were three main changes in the CCPA's approach to computer program-related claims. One was a modification to the second step of the *Freeman* test that the CCPA had been using to judge the patentability of program-related claims. A second was an increasing reliance on printed output as a way to

---

automatic negative answer to the § 101 question simply because a computer program is involved.

Id. at 811-12 (emphasis added).

It is an error to assume, as the CCPA did in *Bradley*, that merely because the data being manipulated in a programmed digital computer might represent a page from a telephone directory rather than the solution to an equation, the program causing the manipulation to occur or the algorithm underlying it could not be mathematical in character. *See infra* notes 389-94 and accompanying text. The CCPA seems to have confused what is mathematical with what is numerical. As shall be seen, they are not the same thing.

[221] *See, e.g., In re* Sarkar, 588 F.2d 1330, 1332-33 (C.C.P.A. 1979) ("Judicial exclusion of an invention from the 'any process' provision by Congress, and its consequent relegation to the caves of secrecy (as here) must be approached with great care lest the intent of Congress and the constitutionally stated purpose it has implemented be thwarted."). In a footnote, the CCPA asserted that "[t]he growth of technological progress is slowed, not only by judicial restriction to familiar areas, but by myopia." *Id.* at 1332 n.7; *see also In re* Gelnovatch, 595 F.2d 32, 42-48 (C.C.P.A. 1979) (Markey, J., dissenting) (complaining that the Patent Office was avoiding performing its "vital service" of "encouraging true progress in the useful arts" and urging that the Office abandon its attack on program-related patent claims).

[222] *See, e.g., Sarkar*, 588 F.2d at 1333 (*any* process is patentable unless it falls within an already established category of judicially determined exceptions, such as that for mathematical algorithms). In this way, the CCPA attempted to close off any avenue of attack on program-related patent claims that might be based on the fact that the claims were for data processing, data representation, or some other arguably unpatentable subject matter. *But see infra* notes 367-76 and accompanying text.

The CCPA's obsession with algorithms also kept it from noticing other potential patentability problems with some program-related inventions. *See, e.g., In re* Maucorps, 609 F.2d 481, 485 (C.C.P.A. 1979) (affirming rejection of claims for an optimization algorithm because they were for an "algorithm in the *Benson* sense," but not noticing that they were also claims for how one should organize and allocate resources in a sales organization, which would seem to run afoul of the "business methods" rule discussed *supra* note 25); *see also In re* Deutsch, 553 F.2d 689 (C.C.P.A. 1977) (claims for a method of calculating costs related to production in industrial plants, raising possible "business method" patentability problem; but all the analytic focus in the case was on "algorithms," so neither the CCPA nor the Patent Office seemed to notice this aspect of the case). See *infra* notes 282-94 and accompanying text for a discussion of patentability questions presented by other optimization algorithms.

distinguish between patentable and unpatentable claims. A third was an increasing emphasis on the traditionally industrial character of certain program-related claims. Each of these shifts is discussed below.

1. Modifying the Freeman *Test*

In virtually all of the cases decided during this period, the CCPA applied the *Freeman* test it had developed in the second post-*Benson* period to judge the patentability of program-related inventions in an unchanged manner.[223] On its face, the *Freeman* test allowed field of use or technological environment limitations to render patentable claims that would otherwise be for an unpatentable algorithm, for such limitations would mean that use of the algorithm would not be "wholly pre-empted."[224] The Supreme Court in *Flook* had made clear that this approach was inconsistent with *Benson*.[225]

It was not, however, until the ninth of the nine cases of this third post-*Benson* period, *In re Walter*,[226] issued while the CCPA's judgment was once again under scrutiny by the Supreme Court as to two of its other program-related decisions,[227] that the CCPA finally made changes in the *Freeman* test.[228] No change was made to the first step, which inquired

---

[223] *See In re* Sherwood, 613 F.2d 809, 817 (C.C.P.A. 1980), *cert. denied*, 450 U.S. 994 (1981); *In re* Maucorps, 609 F.2d 481, 485 (C.C.P.A. 1979); *In re* Phillips, 608 F.2d 879, 883 (C.C.P.A. 1979); *In re* Diehr, 602 F.2d 982, 988 (C.C.P.A. 1979), *aff'd sub nom.* Diamond v. Diehr, 450 U.S. 175 (1981); *In re* Bradley, 600 F.2d 807, 811-12 (C.C.P.A. 1979), *aff'd by an equally divided court sub nom.* Diamond v. Bradley, 450 U.S. 381 (1981); *In re* Johnson, 589 F.2d 1070, 1077-79 (C.C.P.A. 1978) (discussing and applying the two-step *Freeman* test). *In re* Sarkar is the only case decided in this third post-*Benson* phase which makes no mention of *Freeman*. 588 F.2d at 1330.

[224] *See supra* notes 178-88 and accompanying text.

[225] *See supra* notes 206-10 and accompanying text.

[226] 618 F.2d 758 (C.C.P.A. 1980) (affirming rejection of claims for seismic prospecting method and system).

[227] The *Walter* decision was issued on March 27, 1980. The Supreme Court had granted certiorari in the *Bradley* and *Diehr* cases on March 17, 1980.

[228] Even then, the CCPA denied that it was making the change because of anything the Court said in *Flook*. 618 F.2d at 766-67. But see *supra* notes 206-10 and accompanying text for the *Flook* decision's implicit critique of the *Freeman* test.

Much as the CCPA might criticize the Patent Office for its insubordinate attitude toward the CCPA's decisions, the CCPA was itself rather insubordinate toward the Supreme Court decisions. Thus, it was prudent for the CCPA to modify the *Freeman* test, implicitly criticized in *Flook*, while the Supreme Court had two other CCPA computer program decisions before it, lest some members of the Court be swayed in this new round of cases by annoyance at the CCPA's refusal to heed the

whether the claims recited a "mathematical algorithm."[229] *Freeman*'s second step had been to ask whether granting a patent on the claims would "wholly pre-empt" use of the algorithm; *Walter* changed this step so that "[i]f it appears that the mathematical algorithm is implemented in a specific manner to define structural relationships between the physical elements of the claim (in apparatus claims) or to refine or limit claim steps (in process claims), the claim being otherwise statutory, the claim passes muster under § 101."[230] Thereafter, the CCPA used the *Freeman-Walter* test to review patentability matters, and this test continues to inform the Patent Office's policy.[231] Whether *Walter*'s modification of the *Freeman* test was a really substantive change is debatable,[232] but at least the CCPA

Court's directives.

[229] *Walter*, 618 F.2d at 767.

[230] *Id.* The *Walter* decision went on to say that if the claim presented an algorithm and solved it, but did not apply it to physical elements or process steps, then no amount of post-solution activity or field of use limitation would save it. *Id.* In *In re* Abele, 684 F.2d 902, 906-07 (C.C.P.A. 1982), the patent applicant complained that *Walter*, even with this proviso, did not provide very clear guidance about what would satisfy this second step. For a discussion of a recently issued computer program-related patent raising claim limitations questions, see *infra* note 290 and accompanying text.

[231] *PTO Report*, *supra* note 2, at 565-68. This report tries to articulate what things will and will not satisfy this second step. *Id.* Subsection (a) of the report's section on the *Freeman-Walter* second step discusses "post-solution activity" and gives some examples from the case law of post-solution activity that had satisfied the second step. Subsection (b) discusses "field of use limitations" and indicates that these do not satisfy the second step. Subsection (c) discusses "data gathering steps" and tries to distinguish between those that merely determine values for making calculations (which do not satisfy the *Freeman-Walter* second step) and those in which the claim " 'presents data gathering steps not dictated by the algorithm then but by other limitations which require certain antecedent steps.' " *Id.* at 568 (quoting *In re* Abele, 684 F.2d 902, 908 (C.C.P.A. 1982)). The Patent Office considers the latter category to satisfy the *Freeman-Walter* second step, but the author is unable to comprehend what constitutes this category of data gathering steps and why they would make a claim patentable. Subsection (d) discusses steps which involve the "transformation of something physical." Subsection (e) discusses "structural limitations in process claims" and states that while they are not improper "they are usually not entitled to patentable weight unless they somehow affect or form an essential part of the process." *PTO Report*, *supra* note 2, at 568. This subsection gives three examples of claims reciting structural limitations. In all three cases (one of which was the "reentrant shift register" reference in Benson's eighth claim), the courts decided that the structural limitations did not make the claims patentable (although the Report does not directly say this and might give an untutored reader the sense that these limitations had sufficed). The Report gives no examples where structural limitations in methods claims were successful in satisfying the *Freeman-Walter* second step.

[232] The CCPA denied that *Walter*'s modification of the *Freeman* test marked a substantive change in policy. *Walter*, 618 F.2d at 767.

Walter's description of the second step, as it relates to apparatus claims, seems to be derived from Mackay Radio & Tel. Co. v. Radio Corp. of America, 306 U.S. 86, 94, *reh'g denied*, 306 U.S. 618 (1939), which upheld the patentability of an antenna, the physical structure of which was determined by use of an equation. How the rationale of *Mackay* can be carried over to program innovations

finally moved away from the "wholly pre-empt" language which, under *Flook*, was clearly too narrow an interpretation of *Benson*.

2. *Printed Output As a Distinguishing Feature*

Another way the CCPA came to distinguish between unpatentable and patentable program-related claims during this period was in looking to the production of printed output as a post-solution step that would save the claims from being found unpatentable under *Benson*.[233] This was typified in *In re Johnson*[234] and *In re Walter*,[235] both of which involved claims for improved methods of computerized seismic analysis. Some claims in both cases recited equations, and some called for cross-correlating seismic signals. In *Walter*, Judge Rich explained that Johnson's claims were patentable because of the new seismic record his process created while Walter's claims were only for improved cross-correlations of the signals.[236]

In another case, *In re Phillips*,[237] the Patent Office argued that claims

---

which, even when claimed in apparatus form, do not relate to *physical* structures, but to the structure of a *writing*, is not clear.

Even less clear is how a "refinement" or "limitation" on process claim steps, which the CCPA endorsed in *Walter* as an alternative to *Freeman*'s second step for process claims, would substantively differ from either field of use limitations, or other kinds of limitations or refinements that would mean that use of an equation would not be wholly preempted (which was the essence of the *Freeman* second step). One recent Federal Circuit decision can be read as suggesting that mentioning "read only memory" somewhere in the claims might be enough of a limitation. *See In re* Iwahashi, 888 F.2d 1370, 1373-75 (Fed. Cir. 1989); *see infra* note 290.

[233] *See, e.g., In re* Walter, 618 F.2d at 770 (distinguishing unpatentability of Walter's claims from those in *Johnson* based on lack of physical output recited in Walter's claims); *In re* Sherwood, 613 F.2d 809, 818-19 (C.C.P.A. 1980) (cross-sectional map rendered Sherwood's claims for improved seismic analysis method patentable); *In re* Phillips, 608 F.2d 879, 882-83 (C.C.P.A. 1979) (claim for algorithm for computerized production of architectural specifications patentable); *In re* Johnson, 589 F.2d 1070, 1079-80 (C.C.P.A. 1978) (although some claims involved computations and mathematical formulae, the computing steps were part of a process for producing a new seismic tracing record; because the computations were not the whole of the claimed invention, the claims were not problematic under *Benson* and *Flook*).

[234] 589 F.2d 1070 (C.C.P.A. 1978).

[235] 618 F.2d 758 (C.C.P.A. (1980).

[236] *Id.* at 770.

[237] 608 F.2d 879 (C.C.P.A. 1979). Earlier that year, the CCPA remanded the *Phillips* case to the Patent Office Board of Appeals for a more detailed analysis of the claims. Phillips' application had been rejected at first on the grounds that the invention was algorithmic in character, and hence unpatentable under *Benson*, and that the application made an inadequate disclosure. *See In re* Phillips, 593 F.2d 1021 (C.C.P.A. 1979); *Phillips*, 608 F.2d at 881-82 (discussion of the Board's first *Phillips*

for a method of computerized production of specifications for architectural plans were unpatentable because the process consisted only of the algorithm, data gathering steps, and a printout of results. The Office argued that these were not the kinds of pre- or post-solution activities that would support patentability of a program-related invention under *Flook*.[238] As in some earlier cases, the CCPA in *Phillips* managed to avoid addressing these issues by taking shelter in the two-step *Freeman* test. Finding no "algorithm in the *Benson* and *Flook* sense" (that is, no equations) in the claims, the CCPA perceived no problem in issuing a patent on this invention.[239]

3. *A New Emphasis on the Industrial Character of Program-Related Claims*

In addition to modifying the second step of the *Freeman* test, the CCPA made another, and perhaps more important, effort to appear to be more conciliatory toward Supreme Court rulings in *Benson* and *Flook* during this third post-*Benson* period. Increasingly, the court emphasized the "industrial" or "transformative" character of program-related process claims under review.

Two CCPA decisions typical of this new emphasis are the *Diehr* and *Bradley* cases. Diehr's process, for example, was frequently characterized as a method of curing rubber.[240] Bradley's microcode-related invention was characterized as a new combination of tangible hardware elements,[241]

---

opinion). On remand, the Board adhered to its earlier ruling on Phillips' application. The CCPA then reversed the Board's ruling on Phillips' claims, on both subject matter and disclosure grounds.

[238] *Phillips*, 608 F.2d at 882. There is no mention in the *Phillips* case of the "printed matter" rule, or of the interesting problem of who might own the intellectual property rights (if any) to the generated output. As to the latter, see generally Samuelson, *Allocating Ownership Rights in Computer-Generated Works*, 47 U. PITT. L. REV. 1185 (1986).

[239] *Phillips*, 608 F.2d at 883. See *supra* notes 186-88 and accompanying text concerning earlier cases in which the CCPA avoided difficult subject matter questions by use of the *Freeman* test. But see *infra* notes 367-76 and accompanying text for a discussion of more recent cases that seem to reopen the question whether data processing innovations are patentable in character.

[240] *In re* Diehr, 602 F.2d 982, 988 (C.C.P.A. 1979) ("The claims here on appeal are fundamentally different from the claims involved in *Flook*. They recite a process involving the manipulation of apparatus resulting in the chemical and physical change of starting material . . . ."), *aff'd sub nom.* Diamond v. Diehr, 450 U.S. 175 (1981).

[241] *In re* Bradley, 600 F.2d 807, 812 (C.C.P.A. 1979), *aff'd by an equally divided court sub*

even though Bradley's claims seemed to be for a new data structure for a microprogram function.[242] Such characterizations, which gave the claims the flavor of traditional industrial processes and hardware, enhanced the likelihood of getting positive review from the Supreme Court. Even unsuccessful claimants tended to pick up on this theme in the third post-*Benson* period and made efforts to characterize their inventions as "transformative"[243] or technological in nature.[244] One claimant tried to make his claim appear more technological by drafting his claims in apparatus form.[245]

That there was a tendency in this last set of CCPA decisions before the Supreme Court's *Diehr* decision toward more emphasis on program-re-

---

nom. Diamond v. Bradley, 450 U.S. 381 (1981). Bradley's claims were drafted in apparatus form. The invention related to the use of a firmware module to store soon to be processed data in scratch pad registers rather than in main memory. Storing data in scratch pad registers was said to enhance the efficiency of the computer's execution of programs because of the shorter distance between the register and the central processing unit ("CPU") as compared with main memory and the CPU. *Id.* at 808-09. The CCPA said that it did not need to address the question of whether firmware was patentable, an issue raised by the Patent Office, because the claims, although mentioning firmware, were not *for* firmware. The CCPA said they were instead for a new combination of hardware elements. *Id.* at 810-12.

[242] The examiner rejected the claim because it was for a data structure or an algorithm. *Id.* at 809. Bradley asked for reconsideration on the grounds that he was claiming a data structure in hardware and not a mathematical algorithm. *Id.* at 809-10.

The CCPA opinion does not really discuss the data structure aspect of the case, but only notes that the claimed data structure "is merely that which results from the arrangement of the recited hardware elements in the claimed manner." *Id.* at 812. The CCPA found no difference "between appellants' claimed invention and a strictly mechanical adding machine, which is certainly statutory if claimed in a manner which does not embrace any particular calculation that the machine is capable of making." *Id.* Using the two-step *Freeman* test, the CCPA could find no *Benson* problem with Bradley's invention because his claims did not recite a mathematical algorithm. *Id.* at 813.

[243] *See, e.g., In re* Walter, 618 F.2d 758, 770-71 (C.C.P.A. 1980) (rejecting argument that Walter's improved seismic analysis method involved the transformation of matter); *see also In re* Deutsch, 553 F.2d 689, 692 (C.C.P.A. 1977) (characterizing claims for an optimization algorithm as industrial in character).

[244] *See, e.g., In re* Gelnovatch, 595 F.2d 32, 35-36 (C.C.P.A. 1979) (arguing unsuccessfully that the technological application of his mathematical modeling method for microwave circuit design made his claims patentable); *In re* Sarkar, 588 F.2d 1330, 1330-31 (C.C.P.A. 1978) (arguing unsuccessfully that technological applications of his mathematical model, such as use in designing bridges over bodies of water, rendered the claim patentable).

[245] *See In re* Maucorps, 609 F.2d 481 (C.C.P.A. 1979) (attempting to characterize an invention as technological by making apparatus claims for an algorithm that optimizes the allocation of resources in sales organizations; claims held unpatentable). But for analysis of other optimization algorithm patents and apparatus claims for business functions, see *infra* notes 282-94, 378-86, and accompanying texts.

lated inventions as parts of traditional industrial processes and hardware elements may have some bearing on the interpretation that should be given to the *Diehr* decision itself. *Diehr* can be interpreted as just another instance of the old patent rule that only processes that transform matter are patentable. However, some have interpreted *Diehr* to endorse a rule that claims involving unapplied algorithms represent the only kinds of program-related claims to present any subject matter patentability problems.[246] Depending on the breadth one ascribes to *Diehr*, patents are available now either for all computer program-related inventions or only for those which are elements of more traditional industrial processes.

Given that the *Diehr* Court was deeply divided, that only the defection of two members of the *Flook* majority in combination with the three dissenters in *Flook* permitted the Diehr application to be upheld on subject matter grounds, that three of the four dissenters in *Diehr* are still on the Court, and that three of the five Justices who voted in favor of the Diehr patent have left the Court, it would seem to be more prudent to interpret *Diehr* in a narrower fashion.[247] In light of these facts, it is interesting that the CCPA in the post-*Flook* period came to put considerably more emphasis on the industrial nature of the computer program-related inven-

---

[246] The seeds of the post-*Diehr* "application of an algorithm" standard of patentability were already apparent in several cases decided in the third post-*Benson* phase. *See, e.g., In re* Sarkar, 588 F.2d 1330 (C.C.P.A. 1978). In *Sarkar*, the CCPA posed the issue as if all that mattered was that a claimed invention be an application of an algorithm, rather than a claim for an algorithm itself. *Id.* at 1332. However, Sarkar did not persuade the CCPA that his claim was only for an application of the algorithm. The *Sarkar* opinion suggests that the CCPA still was not convinced by the *Benson* and *Flook* holdings, for the CCPA said, "Consideration of whether the substitution of specific values is enough to convert the disembodied ideas present in the formula . . . into an application of the formula[] is foreclosed by the current state of the law." *Id.* at 1335.

See also In re Walter, 618 F.2d 758 (C.C.P.A. 1980). In *Walter*, Judge Rich set forth the theory that any "applied" algorithm is patentable:

> [I]f an inventor succeeds in applying [a] scientific truth [or mathematical algorithm expressing such a truth] in a specific manner, resulting in the improvement of a process, machine, manufacture, or composition of matter, his invention is statutory, subject to the caveat that the underlying subject matter which has been improved is itself within the bounds of § 101.

Id. at 766. This rationale would seem to make Benson's algorithm not patentable, for the CCPA itself had observed that use of his algorithm permitted the computer to operate faster.

[247] The majority in *Diehr* consisted of Justices Rehnquist (author of the *Diehr* opinion), Burger, Stewart, White, and Powell. Justices Stevens (author of the dissenting opinion), Brennan, Marshall, and Blackmun dissented. *See* Diamond v. Diehr, 450 U.S. 175 (1981). The majority in *Parker v. Flook* consisted of Justices Stevens (author of the *Flook* opinion), Brennan, White, Marshall, Blackmun, and Powell. Justices Stewart (author of the dissent), Burger, and Rehnquist dissented. *See* Parker v. Flook, 437 U.S. 584 (1978).

tions that came before it. As the discussion of *Diehr* in Part IV of this Article demonstrates, one can assume that if *Diehr* and *Bradley* had not appeared to involve such seemingly traditionally patentable subject matter, the Court might well have ruled the other way.

V. *DIAMOND V. DIEHR* AND ITS AFTERMATH: ARE ALL PROGRAM-RELATED INVENTIONS NOW PATENTABLE?

In March 1981, the Supreme Court upheld the patentability of a computer program-controlled process for the curing of rubber in molds in *Diamond v. Diehr*.[248] In the year or so after the *Diehr* decision, there were four additional CCPA decisions on the patentability of computer program-related inventions. These decisions had already begun the appeal process before the *Diehr* decision came down.[249] For seven years thereafter, however, no Court of Appeals decision on the patentability of computer program-related inventions occurred.[250] It was not until 1989 that

---

[248] 450 U.S. 175 (1981).

[249] *In re* Taner, 681 F.2d 787 (C.C.P.A. 1982) (upholding patentability of method of seismic exploration involving simulation of seismic waves); *In re* Abele, 684 F.2d 902 (C.C.P.A. 1982) (upholding patentability of some claims and denying it as to others in CAT-scan related improvement); *In re* Pardo, 684 F.2d 912 (C.C.P.A. 1982) (upholding patentability of a compiler program); *In re* Meyer, 688 F.2d 789 (C.C.P.A. 1982) (denying patentability of medical diagnosis expert system). The *Meyer* case is something of an aberration from the CCPA's usual approach to computer program-related inventions and will be discussed at length *infra* notes 363-73 and accompanying text.

Pardo's patent (No. 4,398,249, issued 13 years after the application was filed) is the basis of the patent infringement lawsuit filed in 1989 against Lotus Development Corp., among others, which claims that Pardo was the first inventor of an important element of spreadsheet programs, a natural order recalculation function. (To read the *Pardo* opinion, one would think that the claims were for a compiler.) *See A "White Knight" Draws Cries of "Patent Blackmail,"* N.Y. Times, Jan. 14, 1990, § 3, at 5, col. 1. While the current lawsuit is against software developers, customers of potentially infringing spreadsheet programs also have reason to worry, for they, as unauthorized users of the recalculation function, also could be held liable for patent infringement.

[250] Paine, Webber, Jackson & Curtis, Inc. v. Merrill Lynch, Pierce, Fenner & Smith, Inc., 564 F. Supp. 1358 (D. Del. 1983), is a federal district court decision concerning the patentability of a program-related invention that occurred in the seven years after the *Meyer* decision. *See infra* notes 378-86 and accompanying text. Since the *Diehr* decision, there have been a few additional patent cases involving program-related inventions. *See, e.g.*, White Consol. Indus. v. Vega Servo-Control, Inc., 713 F.2d 788 (Fed. Cir. 1983) (holding patent for computerized control of machine tooling equipment invalid for failure to meet enablement requirement); Safe Flight Instrument Corp. v. Sundstrand Data Control, Inc., 706 F. Supp. 1146 (D. Del. 1989) (containing some discussion of the patentability of invention in finding no infringement of patent by maker of computerized equipment for avionics system detection of windshear), *aff'd*, 899 F.2d 1228 (1990); Wentworth v. Gulton Indus., 578 F. Supp.

the Court of Appeals for the Federal Circuit handed down its first two of such decisions, *In re Grams*[251] and *In re Iwahashi*.[252] The court affirmed the Patent Office's rejection of program-related claims in *Grams* [253] and overturned the Patent Office's ruling in *Iwahashi*.[254] If one compares the lack of decisions in the nine years following the *Diehr* decision with the twenty decisions on program-related inventions in virtually the same time span between the Supreme Court's *Benson* and *Diehr* decisions,[255] it is clear that something must have changed.

This change seems to be in the Patent Office's policy and practice regarding program-related inventions.[256] After *Diehr*, the Patent Office ceased to resist issuing patents for computer program-related inventions. It now only rarely rejects applications pertaining to those inventions on sub-

---

508 (N.D. Tex. 1982) (holding patent for computerized sign display invalid on obviousness grounds), *aff'd*, 722 F.2d 1253 (5th Cir. 1984), *cert. denied*, 469 U.S. 817 (1984).

[251] 888 F.2d 835 (Fed. Cir. 1989).

[252] 888 F.2d 1370 (Fed. Cir. 1989).

[253] *Grams*, 888 F.2d 835 (affirming rejection of claims for a method of diagnosing the existence of an abnormal condition in complex systems on the ground that it involved a mathematical algorithm); *see infra* notes 374-77 and accompanying text.

[254] *Iwahashi*, 888 F.2d 1370 (ruling in favor of a patent for an apparatus "unit" that improved calculation of correlation coefficients for use in pattern recognition because the claim made reference to a "unit" and to a "read only memory"); *see infra* note 294.

[255] Eighteen of these 20 decisions were issued in the five-year period between the Supreme Court's opinions in Dann v. Johnston, 425 U.S. 219 (1976), and Diamond v. Diehr, 450 U.S. 175 (1981).

[256] *See PTO Report*, *supra* note 2 (concerning the Patent Office's present view of its policy and practice); *see also Samuels Testimony*, *supra* note 2, at 7. It is, however, difficult to discern exactly what will pass muster from the Report. Patents, such as those issued to Karmarkar and Bracewell, *see infra* notes 282-94 and accompanying text, and those issued for expert systems and other data processing methods, *see infra* notes 350-51 and accompanying text, do not seem to be consistent with the Report, the *Benson*, *Flook*, and *Diehr* line of cases, or the *Meyer* and *Grams* decisions. *See infra* notes 347-77 and accompanying text.

The clearest operative distinction currently employed in the Patent Office is between "computer programs" (which are said not to be patentable) and "computer processes" (which are considered to be patentable if other patent requisites are met). *PTO Report*, *supra* note 2. The Report does not explain why programs themselves are not patentable, and indeed there do seem to be some patents that have issued for programs per se. *See, e.g.*, Patent No. 4,566,066 (securities valuation system).

In response to a letter from Congressman Kastenmeier, chair of the House Judiciary Committee's Subcommittee on Courts, Intellectual Property and the Administration of Justice, the Acting Commissioner of Patents and Trademarks indicated that the 1989 Guidelines reflect the policy followed by the Patent Office since 1982 in reviewing applications involving mathematical algorithms and computer processes. Ratner & Nigon, *The Patentability of Computer Programs: The PTO Guidelines*, In re Grams, *and* In re Iwahashi, COMPUTER LAW., Dec. 1989, at 21, 21.

ject matter grounds.[257] Though it has taken some years for the message to reach the software industry, patent lawyers no longer look upon the Patent Office as nay-sayers when they present claims for program-related inventions.[258] The Patent Office seems to have adopted a liberal, rather than a conservative, interpretation of *Diehr*. It is issuing patents for algorithms and a wide range of other software-related innovations,[259] in contrast to its former practice of issuing patents to traditional industrial or physical processes utilizing computer programs as elements. To judge whether the Patent Office's more expansive view of the patentability of program-related inventions is warranted, the *Diehr* decision must be reviewed.

A. *The Supreme Court's Decision in* Diehr

From the opening sentence of Justice Rehnquist's opinion in the *Diehr* case, the industrial nature of the Diehr process is repeatedly hammered home. The first sentence states that the Court granted certiorari "to determine whether a process for curing synthetic rubber which included in several of its steps the use of a mathematical formula and a programmed digital computer is patentable subject matter under 35 U.S.C. § 101."[260] The last sentence of the opinion states that the claims were not for a mathematical formula, but were "drawn to an industrial process for the molding of rubber products."[261] Between these two statements, additional

---

[257] *Ex parte* Murray, 9 U.S.P.Q.2d (BNA) 1819 (PTO Bd. App. 1988), demonstrates a rare instance of a rejection of a program-related claim between *Meyer* and *Grams*.

[258] *See, e.g.*, Sumner & Lundberg, *supra* note 11, at 1-3 (contrasting the "old" patent climate with the "new" patent climate).

[259] For a discussion of "mathematical algorithm" patents, see *infra* notes 282-95 and accompanying text. For examples of expert system and other data processing patents, see *infra* notes 350-51.

[260] *Diehr*, 450 U.S. at 177.

[261] *Id.* at 192-93. Diehr's "invention" probably should have been rejected as too obvious to be patentable. The Arrhenius equation (by which the time for curing the rubber was calculated) was already in the state of the art. If the problem with getting a perfect cure was in obtaining more continuous calculations than a human engineer could obtain, then computerizing the calculation process was the obvious choice. Continuous recalculation is precisely the kind of task computers perform well. Justice Rehnquist took note of this, for late in the opinion he observed that it might be determined later that no patent should have issued because of the obviousness or lack of novelty of the claimed invention. *Id.* at 191. Justice Stevens' dissent, however, points out that the Diehr application had been rejected solely on subject matter grounds. *Id.* at 211 n.33. Hence, if the CCPA's ruling were affirmed, the patent would issue.

emphasis is placed on the industrial nature of Diehr's process,[262] a characterization that the dissent found difficult to accept.[263]

Although the *Diehr* opinion makes a few broad statements concerning patentable subject matter,[264] it is more circumspect when delving into the

---

[262] The first section of the Court's opinion, for example, describes Diehr's contribution to the art of rubber curing. The problem with synthetic rubber curing processes before Diehr's invention was that rubber tended to be over- or undercured. Diehr claimed that his process "ensures the production of molded articles which are properly cured." *Id.* at 177. Diehr's contribution to the art of rubber curing involved "the continuous measuring of the temperature inside the mold cavity, the feeding of this information to a digital computer which constantly recalculates the cure time, and the signaling of the computer to open the press." *Id.* at 179.

[263] Justice Stevens, in his dissenting opinion in *Diehr*, asserted that the majority's ruling "rests on a misreading of the Diehr and Lutton patent application." *Id.* at 194. Justice Stevens pointed out that:

> [Their application] teaches nothing about the chemistry of the synthetic rubber-curing process, nothing about the raw materials to be used in curing synthetic rubber, nothing about the equipment to be used in the process, and nothing about the significance or effect of any process variable such as temperature, curing time, particular compositions of material, or mold configurations. In short, Diehr and Lutton do not claim to have discovered anything new about the process for curing synthetic rubber.

Id. at 206.

Justice Stevens gave three reasons why he could not accept the majority's characterization of the claims:

> First, there is not a word in the patent application that suggests that there is anything unusual about the temperature-reading devices used in this process — or indeed that any particular species of temperature-reading device should be used in it. Second, since devices for constantly measuring actual temperatures — on a back porch, for example — have been familiar articles for quite some time, I find it difficult to believe that a patent application filed in 1975 was premised on the notion that a "process of constantly measuring the actual temperature" had just been discovered. Finally, the Patent and Trademark Office Board of Appeals expressly found that "the only difference between the conventional methods of operating a molding press and that claimed in [the] application rests in those steps of the claims which relate to the calculation incident to the solution of the mathematical problem or formula used to control the mold heater and the automatic opening of the press." This finding was not disturbed by the Court of Customs and Patent Appeals and is clearly correct.

Id. at 207-08 (quoting Diehr's patent application). Justice Stevens concluded that what Diehr and Lutton "claim to have discovered, in essence, is a method of updating the original estimated curing time by repetitively recalculating that time pursuant to a well-known mathematical formula in response to variations in temperature within the mold." *Id.* at 209. This made it "strikingly reminiscent of the method of updating alarm limits that Dale Flook sought to patent." *Id.* Justice Stevens regarded Diehr's invention to be as unpatentable as Flook's invention had been.

[264] The *Diehr* opinion, for example, cautions against reading limitations into the patent law that the legislature had not expressed, and it quotes a statement from the legislative history to the 1952 Patent Act "which informs us that Congress intended statutory subject matter to 'include anything under the sun that is made by man.' " *Id.* at 182. This is a very broad interpretation of patentable

patent meaning of the term "process." The opinion quotes at length from *Cochrane v. Deener*,[265] on which the Patent Office had long relied for its narrow interpretation of the term "process" in relation to computer programs.[266] It also cites *Benson* approvingly when it speaks of "transformation" as " 'the clue to the patentability of a process claim that does not include particular machines.' "[267]

Turning its attention to Diehr's claims and analyzing them in light of the Court's long history of interpreting processes under *Cochrane*'s transformation standard, the opinion states that a majority of the Court "think[s] that a physical and chemical process for molding precision synthetic rubber products falls within the § 101 categories of possibly patentable subject matter. . . . Industrial processes such as this are the types which have historically been eligible to receive the protection of our patent laws."[268]

The majority's conclusion about the patentability of Diehr's process was not altered by the fact that some steps of the process involved a mathematical formula or a programmed digital computer.[269] The opinion contrasts the claims in *Benson* and *Flook* with Diehr's claims: the Benson and Flook claims had been for mathematical formulae, whereas Diehr's claim was for a rubber curing process.[270] The opinion emphasizes that Diehr's process used a long-known equation and made no claim to preempt use of it. Instead, Diehr was "seek[ing] only to foreclose from others the use of that equation in conjunction with all of the other steps in their claimed process."[271]

---

subject matter. *See infra* notes 422-25 and accompanying text.

[265] 94 U.S. 780 (1877), *cited in Diehr*, 450 U.S. at 182-84. The Court described *Cochrane* as "defining the nature of a patentable process" and then went on to quote the *Cochrane* definition as it had done in *Benson*. *Diehr*, 450 U.S. at 182-83; *see supra* notes 102-05 and accompanying text.

[266] *See supra* notes 46-47 and accompanying text.

[267] *Diehr*, 450 U.S. at 184 (quoting Gottschalk v. Benson, 409 U.S. 63, 70 (1972)).

[268] *Id.* In a footnote, the Court observed that rubber curing processes had long been regarded as patentable, quoting from yet another nineteenth-century case. *Id.* at n.8. See *supra* note 263 and 450 U.S. at 181 for alternate characterizations of the Diehr invention.

[269] *Diehr*, 450 U.S. at 185.

[270] *Id.* at 185-87.

[271] *Id.* at 187. The *Diehr* interpretation of *Benson* as a "total preemption" case parallels earlier interpretations of *Benson*. *See supra* notes 181-83 and accompanying text. Had the Court's *Diehr* decision been unanimous or nearly so, one might have wondered whether the Court in 1981 would have answered the question posed in *Flook* differently as to whether the Pythagorean theorem would be patentable as long as the discoverer limited the scope of his or her claims to the art of surveying.

The *Diehr* majority opinion observes, as had *Benson* and *Flook*, that the discovery of laws of nature, natural phenomena, and mathematical formulae are not patentable discoveries.[272] The *Diehr* opinion does not really explain the reasons for this rule. However, it appears from the context of the Court's discussion of this issue that such discoveries are considered unpatentable partly because of their "abstractness"[273] and partly because of the Court's view that such discoveries are merely recognizing what was already in existence, rather than creating something new.[274] The opinion contrasts the unpatentability of a discovery of a law of nature or mathematical equation to the patentability of "an *application* of a law of nature or mathematical formula to a known structure or process. . . ."[275]

---

[272] *Diehr*, 450 U.S. at 185. Justice Rehnquist went on to say that *Benson* and *Flook* "stand for no more than these long-established principles." *Id.*

[273] *Id; see also infra* note 278.

[274] The opinion, for example, links the discovery of natural phenomena and the discovery of equations. *Diehr*, 450 U.S. at 186. Even though there might be considerable human ingenuity in discovering some natural phenomenon, such a discovery is thought not to be patentable because the phenomenon has always had the beneficial characteristic just discovered. *See* Funk Bros. Seed Co. v. Kalo Inoculant Co., 333 U.S. 127 (1948). Patentability arises when there has been some human "tinkering" with nature to create a new phenomenon.

The issue of whether equations or laws of nature are "discovered" rather than "invented" is a deep philosophical question. It has pervaded the Supreme Court case law on the patentability of program-related innovations. *See supra* notes 99, 204, and accompanying texts. Scientists, mathematicians, and philosophers are split over this issue. *See supra* note 99. One view is stated in Martin Gardner's preface to a recently published book: "When a physicist or a mathematician experiences a sudden 'aha' insight, Penrose believes, it is more than just something 'conjured up by a complicated calculation.' It is mind making contact for a moment with objective truth." Gardner, *Foreword* to R. PENROSE, *supra* note 99, at vi.

It quite obviously makes no sense to make the patentability of mathematical formulae turn on whether they are "invented" or "discovered," for it is impossible to know for certain which is the case.

[275] *Diehr*, 450 U.S. at 187. It is chiefly this statement from *Diehr* that has caused some lawyers to interpret *Diehr* as making only "unapplied" equations unpatentable. *See, e.g.*, Sumner & Lundberg, *supra* note 11.

The *Diehr* majority opinion makes one arresting and unexplained statement: "Arrhenius' equation is not patentable in isolation, but when a process for curing rubber is devised which incorporates in it a more efficient solution of the equation, that process is at the very least not barred at the threshold by § 101." *Diehr*, 450 U.S. at 188. What is especially confusing and troubling about this statement is its emphasis on efficient solutions to equations as an apparently validating factor for patentability. What made Diehr's use of the Arrhenius equation "more efficient" was that it was computerized.

This statement seems difficult to reconcile with both *Flook*'s interpretation of *Benson* and with CCPA cases such as *Waldbaum II*, 559 F.2d 611 (C.C.P.A. 1977), *see supra* note 175, which say that merely limiting use of an equation to a particular field of application or technological environment is insufficient to save it from *Benson*'s proscription. Although Justice Rehnquist demonstrated his willingness to change this rule when he joined the *Flook* dissent, it seems unlikely that his col-

The *Diehr* opinion ends with a discussion of two additional issues. First, the Court considered whether claims should be analyzed "as a whole" (as the CCPA contended) or dissected to determine what "points of novelty" they contain (as the Patent Office contended).[276] The majority concluded that the CCPA's approach was more appropriate than the Patent Office's.[277] Second, the Court discussed what the proper procedure should be in analyzing claims like Diehr's that involved an equation or computer program as elements.[278] The opinion cites *Flook* approvingly for

---

leagues in the slender majority in *Diehr* perceived themselves to be overturning *Flook* in this respect. Moreover, later in the opinion, Rehnquist disclaimed that *Diehr* goes this far. *See infra* note 279 and accompanying text.

[276] *See supra* notes 189-208 and accompanying text.

[277] In a long footnote, the *Diehr* opinion criticizes the "view" that *Flook* calls either for dissection of claims into new and old elements or for the assumption that any algorithm claimed as part of the process was already in the state of the art when judging the subject matter of the claims. 450 U.S. at 189 n.12. *But see supra* notes 193-208 and accompanying text.

Two issues deserving separate attention are, first, whether claims should be dissected into new and old elements, and second, whether one should treat the algorithm or calculation steps as already in the state of the art when judging whether a claim is for patentable subject matter.

As to the first issue, it is clear that the "point of novelty" test did involve some dissection of claims into new and old elements. *See supra* notes 193-208 and accompanying text. It did so out of concern that otherwise someone could evade the rule against patenting algorithms by tacking on some conventional additional step to the claim. *See supra* notes 207-08 and accompanying text. Furthermore, in *Diehr*, the court said that "insignificant" post-solution activity would not save a claim. *See infra* note 279. This would seem to call for dissection of the claims and assessment of the weight to be given to their elements. (Why could not "significance" turn, in part, on whether the element was conventional?).

Justice Stevens questioned the use of a "significance" test. In both *Flook* and *Diehr*, he pointed out, "the post-solution activity is a significant part of the industrial process. But in neither case should that activity have any *legal* significance because it does not constitute a part of the inventive concept that the applicants claimed to have discovered." *Diehr*, 450 U.S. at 215 (emphasis in original).

Assuming that the algorithm already existed in the state of the art was a more controversial aspect of *Flook*. *See* Parker v. Flook, 437 U.S. 584, 591-95 (1978). However, *Flook* was not the first patent case to use such a test. *See, e.g.*, O'Reilly v. Morse, 56 U.S. (15 How.) 402 (1853). However, the test does seem somewhat unfair, for it would deny a patent to someone who simultaneously discovers both a law of nature and an application of that law to the useful solution of a problem on the ground that if one assumes the law of nature is already in the state of the art, the application of it may be obvious and therefore unpatentable. It could lead to the anomalous result that someone who understood why his or her invention worked would be denied a patent, whereas someone who discovered the same application by serendipity would be eligible for a patent.

Post-*Diehr* claim analysis seems to require some dissection of program-related claims involving algorithms. *Abele* has been construed to call for reading the algorithm out of the claims; if nothing (or nothing of significance) is left in the claims (after the algorithm is read out) that is patentable, the claims are for unpatentable subject matter. *See In re* Abele, 684 F.2d 902, 906-07 (C.C.P.A. 1982); *PTO Report*, *supra* note 2, at 565.

[278] The issue, said the Court, was whether the claim seeks a patent for the formula "in the

the propositions that neither limiting the field of use of an equation nor tacking on "insignificant" post-solution activity[279] would save a claim for an algorithm. These rules were necessary to prevent "artful" claim drafting aimed at evading the rule against patenting algorithms.[280] The majority opinion concludes:

> [W]hen a claim containing a mathematical formula implements or applies that formula in a structure or process which, when considered as a whole, is performing a function which the patent laws were designed to protect (for example, transforming or reducing an article to a different state or thing), then the claim satisfies the requirements of § 101.[281]

Diehr's claim did so and hence was for patentable subject matter.

B. *The Patentability of Algorithms After* Diehr*: The Karmarkar Algorithm*

A *New York Times* article, entitled *Patents on Equations: Some See a Danger*, gives examples of five patents issued between 1987 and 1989 for

---

abstract." *Diehr*, 450 U.S. at 191. *Benson* forbids patents on abstract formulae, and the Court construed *Flook* as holding that "this principle cannot be circumvented by attempting to limit the use of the formula to a particular technological environment." *Id.*

[279] In a lengthy footnote, Justice Rehnquist tried to reconcile *Flook* and *Diehr* concerning "post-solution activity." *Id.* at 192 n.14. Justice Stevens' dissent, however, takes issue with the characterization of the post-solution activity in *Flook* as "insignificant." *Id.* at 215. The post-solution activity in *Flook* (updating the alarm limit) was indeed the sole purpose of the process. It was not the insignificance of Flook's post-solution activity that rendered its addition to the claim insufficient to support the claim, but rather, the fact that this post-solution activity had been in the state of the art for some time. *See supra* notes 201-02 and accompanying text.

[280] Although Justice Rehnquist's majority opinion in *Diehr* gives lip service to the notion that the proscriptions of *Benson* and *Flook* cannot be avoided by clever draftsmanship, *see Diehr*, 450 U.S. at 192, Justice Stevens thought that "the most significant distinction between the invention at issue in *Flook* and that at issue in this case lies not in the characteristics of the inventions themselves, but rather in the drafting of the claims." *Id.* at 210 n.32. The dissent showed how the Diehr claims could be drafted in *Flook*-like language. *Id.* That the Court in *Diehr* was willing to uphold patentability of the Diehr claims seemed to the dissenters to mean that evasion of *Flook*'s proscription by artful drafting was, despite the majority's protestations otherwise, a result which the Court was actually quite willing to accept.

[281] *Id.* at 192. This statement parallels the CCPA's *Freeman-Walter* test of algorithm patentability, differing from it only (though significantly) in its terms of its focus on transformation of matter. *See supra* notes 226-32 and accompanying text.

equations intended to be implemented in computer programs.[282] The article discusses at some length Narendra Karmarkar's patented algorithm (assigned to his employer AT&T Bell Laboratories) for efficient resource allocations.[283] While any of the patents discussed in this article could be used to address the question of whether algorithms of this sort are patentable under *Diehr*, it is useful to focus on one such patent. The Karmarkar patent is a good example because more has been written about it,[284] and it bears some resemblance to the resource allocation algorithm which the CCPA held unpatentable in 1979.[285]

The Karmarkar algorithm addresses a long-standing mathematical problem, frequently referred to as the "traveling salesman" problem. It concerns how to calculate the most efficient route between points x and y when there are such a large number of possible paths that the number of computations required to make the determination is unmanageably large. Although some algorithms that made progress towards solving this problem[286] had been previously discovered, the Karmarkar linear programming algorithm has been widely recognized to be the best solution to date because it is reliable and reduces considerably the number of calculations that have to be made.[287]

There are many applications of the Karmarkar algorithm. One is in the

---

[282] N.Y. Times, Feb. 15, 1989, at D1, col. 4. The five patents were as follows: the 1988 Karmarkar patent for a method of efficient resource allocation (No. 4,744,028), the 1987 Bracewell patent for a system of solving the discrete Bracewell transformation (No. 4,646,256), the 1989 Duhamel patent for a method of performing discrete cosine transformation (No. 4,797,847), the 1988 TRW patent for squared radix discrete Fourier transform algorithm (No. 4,768,159), and the 1989 Eastman Kodak patent for a system incorporating an error tolerant picture compression algorithm (No. 4,797,729).

[283] The patent claims only "industrial applications" of the algorithm, a scope of claims consistent with Professor Chisum's proposal for patenting algorithms. *See infra* notes 337 and accompanying text. Chisum, however, seems to argue that it is necessary to overrule *Benson* to make such claims patentable. It is worth recalling that *Diehr* indicates that mere field of application limitations will not make an algorithm patentable. *See supra* note 279 and accompanying text. For questions about what "industrial applications" might include, see *infra* note 338 and accompanying text.

[284] *See, e.g., Formula Aids Planners at Bell Labs*, N.Y. Times, Dec. 10, 1986, at D8, col. 4; *Breakthrough in Problem Solving*, N.Y. Times, Nov. 19, 1984, at A1, col. 3.

[285] *In re Maucorps*, 609 F.2d 481 (C.C.P.A. 1979) (holding claims for system that optimizes the organization of sales representatives in a business to be for an "algorithm in the *Benson* sense," despite claimant's drafting of claims in apparatus form).

[286] *See, e.g.*, Newell, *supra* note 15, at 1028.

[287] *See, e.g., Simplified Simplex: New Algorithm for Solving Linear Programming Problems for Industrial Applications*, SCI. AM., Jan. 1985, at 54.

planning of airplane flight routes. AT&T has developed the Korbx system, a computer system implementing the Karmarkar algorithm, which AT&T has reportedly made available to the Air Force for $6.5 million.[288] Korbx is especially useful for optimizing resource allocations when the number of variables exceeds 10,000.[289]

There are a number of parallels between the Karmarkar algorithm and the Benson algorithm held unpatentable almost two decades ago. Both are algorithmic advances said to speed up the processing of data in a computer. Both have their only practical utility in a digital computer implementation. Both are unrestricted in their claims as to the machinery or technological field to which they apply.[290] The Karmarkar algorithm calls for numerical calculations even more directly than Benson's algorithm.

Because of these similarities, it would appear that, apart from academic uses of the algorithm which AT&T says it has waived,[291] a patent on the Karmarkar algorithm would, to state the issue in *Benson* terms, effectively preempt use of the algorithm.[292] This suggests that if *Benson* were still the law, the Karmarkar algorithm could not be considered patentable subject matter any more than the Benson algorithm was.

Thus, the following question arises: Did the Supreme Court's decision in *Diehr* change what would otherwise be the clear application of *Benson* to the Karmarkar algorithm? The answer, of course, depends on one's reading of the *Diehr* decision. There is language in *Diehr* that says that only mathematical formulae "in the abstract" are unpatentable, but that an "application" of a law of nature or mathematical formula is patenta-

---

[288] *See, e.g.*, Anthes, *Air Force Unit Receives Korbx Optimization Tool*, FED. COMPUTER WEEK, Feb. 27, 1989, at 10.

[289] *Id.* For example, optimizing the planning of transatlantic flight routes, which involves 70,000 variables and 10,000 constraints, would previously have required two weeks to calculate; using Korbx, it takes only fifteen minutes. *Id.* Korbx is also reported to take only an hour to schedule airlift support aircraft, a problem involving 321,000 variables and 10,000 constraints. *Id.*

[290] While the Karmarkar patent claims only "industrial applications" of the algorithm, *see supra* note 283, this seems tantamount to a claim for all nonacademic applications of the algorithm. Thus, the apparent restriction in the claims may not, in fact, be a meaningful restriction on the scope of the claims.

[291] *See, e.g.*, *Patents on Equations: Some See a Danger*, N.Y. Times, Feb. 15, 1989, at D1, col. 4.

[292] Recall that *Flook* and *Diehr* make clear that mere field of use limitations are not enough to make *Benson* inapplicable. *See supra* notes 207, 279, and accompanying texts.

ble.[293] If one focuses on this language, then *Diehr* can be read to support the patentability of the Karmarkar algorithm. Karmarkar does not claim to own the algorithm "in the abstract," but only in those areas where it is useful in solving real-world resource allocation problems.[294]

If, however, one remembers that *Diehr* was a five-to-four decision, that more dissenters than members of the majority are still on the Court, that the Court in *Diehr* repeatedly emphasized that Diehr's process was traditionally industrial in nature and transformed matter, and that the Court cited *Benson* and *Flook* approvingly throughout the *Diehr* opinion,[295] it is clear that *Diehr* provides a fragile base for supporting the patentability of the Karmarkar algorithm. Unless and until *Benson* is overruled, patents on algorithms such as Karmarkar's are suspect.

This then leads to the question: Should *Benson* be overruled? It is to this question we now turn.

---

[293] *See supra* note 278 and accompanying text.

[294] In contrast to the Karmarkar patent, which is drafted in method form, Ronald Bracewell's improved fast Fourier transform algorithm reportedly was drafted in apparatus form in order to pass muster with the Patent Office. *See* Note, *Computer Intellectual Property and Conceptual Severance*, 103 HARV. L. REV. 1046, 1059 (1990). The question the Bracewell patent raises of whether a ruling on the patentability of an algorithm should turn on the fact that the claim is drafted in apparatus rather than method form is an old one. *See supra* notes 163, 174, and accompanying text.

While the CCPA and the PTO Report both repudiate the idea that patentability should turn on the form in which claims are drafted (*see supra* note 174 and accompanying text; *PTO Report*, *supra* note 2, at 566), the Bracewell patent and the Federal Circuit's recent ruling in *In re* Iwahashi, 888 F.2d 1370 (Fed. Cir. 1989), revive the question whether the form of the claim makes a difference. In *Iwahashi*, the court ruled in favor of a patent on an apparatus claim for a "unit" for improved calculation of correlation coefficients for use in pattern recognition, such as voice recognition, because the claim made reference to a "unit" and to "read only memory" ["ROM"]. Iwahashi asserted that these references demonstrated that the claim contained specific structural limitations in satisfaction of the *Freeman-Walter* test. The Federal Circuit agreed, saying that the "claim as a whole certainly defines [an] apparatus in the form of a combination of interrelated means and we cannot discern any logical reason why it should not be deemed statutory subject matter as either a machine or a manufacture as specified in § 101." *Id.* at 1375. Although after *Iwahashi*, the Patent Office issued a "clarification" stating its view that the Federal Circuit did not mean to make all program-related claims in apparatus form patentable as long as they mention ROMs (*see PTO Says* Iwahashi *Does Not Affect Policy on Patenting Software*, 39 Pat. Trademark & Copyright J. (BNA) 387, 387-88 (1990)), the decision itself does not seem so limited.

In view of *Iwahashi*, it is worth recalling that the CCPA had upheld one of Benson's claims because it made reference to "signals" and "reentrant shift registers." *See supra* note 83. The Supreme Court, however, ruled that both of Benson's claims were for unpatentable subject matter. In *Iwahashi*, the Federal Circuit characterized *Benson* as making unpatentable only claims for "abstract" mathematical formulae and algorithms. *Iwahashi*, 888 F.2d at 1374.

[295] *See supra* notes 267-70 and accompanying text.

C. *Should* Benson *Be Overruled?*

1. *Professor Chisum's Argument*

Professor Chisum presents a forceful and cogent set of arguments for overturning the Supreme Court's *Benson* decision and for recognizing computer program algorithms as an appropriate subject matter for patenting in his article, *The Patentability of Algorithms*.[296] His article offers four main contentions in support of this proposition.

First, Professor Chisum asserts that prior to the Supreme Court's decision in the *Benson* case, the CCPA had "established a firm basis for the patenting of algorithmic ideas" when it rejected "for appropriate reasons" the "mental steps" doctrine.[297] Second, he declares that the *Benson* decision was "poorly reasoned" and explains the result in the case as "stemm[ing] from an antipatent judicial bias that cannot be reconciled with the basic elements of the patent system established by Congress."[298] Third, he states that the "awkward distinctions and seemingly irreconcilable results of the case law since *Benson* . . . are the product of the analytical and normative weakness of *Benson* itself."[299] Fourth, he offers a set of policy arguments in favor of algorithm patents.[300]

Chisum's policy analysis begins with the assertion that since computer program algorithms fall within the literal terms of the patent statute (in other words, algorithms are processes), the burden should be on those who seek to exclude such algorithms from patent protection to make the case against such patents.[301] Chisum asserts that no cogent policy analysis has ever been put forth to support the restricted definition of "process" that *Benson* seems to give it.[302] Algorithmic advances, he points out, can en-

---

[296] Chisum, *supra* note 12.

[297] *Id.* at 961. Professor Chisum's discussion of the pre-*Benson* case law can be found at 963-71. The CCPA's *Musgrave* case repudiates the "mental process" doctrine, on which the Patent Office had relied in many of the early computer program cases. *See supra* notes 27-38 and accompanying text. This case contains the patentability standard of which Professor Chisum approves. For a discussion of *Musgrave*, see *infra* notes 315-28 and accompanying text.

[298] Chisum, *supra* note 12, at 961. Professor Chisum's analysis of the *Benson* decision can be found at 971-92.

[299] *Id.* at 961-62. Professor Chisum's analysis of the post-*Benson* case law can be found at 992-1009.

[300] *Id.* at 962. The policy analysis can be found at 1009-20.

[301] *Id.* at 1011.

[302] *Id.* at 1013.

able computers to be used more efficiently and effectively. Research to develop new algorithms can be expensive, and patent incentives are needed to encourage firms to invest in such research.[303]

2. *A Reply to Professor Chisum*

Professor Chisum correctly states that the pre-*Benson* case law eliminated doctrinal barriers to the patentability of computer program algorithms. This article questions whether the CCPA did so for appropriate reasons and with considered judgment about the consequences likely to flow from this action.[304] While Professor Chisum is also correct that the Supreme Court poorly articulated its rationale in *Benson*, this article argues that there is a more substantial basis in patent law for the Court's decision than Professor Chisum perceives. Unlike Professor Chisum, the author does not see evidence of an "antipatent judicial bias" in *Benson.* Indeed, as she reads *Benson*, the Court was keeping an open mind about the patentability of computer program-related inventions.

Professor Chisum also rightly observes that the post-*Benson* case law is replete with awkward distinctions and results that are difficult to reconcile. This article has, however, shown that much of the muddle in the post-*Benson* decisions was a muddle of the CCPA's own making. That which was not the CCPA's doing was due more to the inability of legal decisionmakers to grasp the nature of computer program innovations than to the reasoning in the *Benson* decision.[305]

Computer scientist Allen Newell responded to the Chisum article by commenting:

> Chisum may well be right that the *Benson* case has brought on much analytic confusion in the software patent area. He may also be right in supposing that, if *Benson* had only been decided differently, the specific confusion that occurred in its aftermath would not have materialized. It seems to be his view that a different holding in *Ben-*

---

[303] *Id.* at 1014-16. Because algorithms are building blocks for making useful products and those products are independently protectible by copyright, Chisum argues that patents on program algorithms would be less burdensome than patents on other technological innovations. In addition, licensing is likely to be the best strategy for optimizing returns on the patent. *Id.* at 1016-17.

[304] *See supra* notes 54-88, *infra* notes 396-409, and accompanying texts.

[305] *See supra* notes 148-244 and accompanying text.

son would have brought about analytic sweetness and light.

> My point is precisely to the contrary. Regardless how the *Benson* case was decided . . . confusion would have ensued. The confusions that bedevil algorithms and patentability arise from the basic conceptual models that we use to think about algorithms and their use. That is why I have entitled my remarks, "The Models Are Broken, The Models Are Broken."[306]

This author joins Professor Newell in questioning whether the model of invention with which the traditional patent system has operated might be "broken" when applied to algorithms and computer programs.[307]

While Professor Chisum discusses some policy reasons for giving patent or patent-like protection to computer program algorithms,[308] he understates the even stronger countervailing policy arguments which are explored at length in Part VII.[309] Even if policy reasons were to favor the patenting of program algorithms, it may be more appropriate for such a decision to be made by Congress than by an appellate court or an administrative agency like the Patent Office because the patenting of computer program algorithms represents a significant departure from patent tradition.[310]

To judge whether *Benson* should be overruled, either legislatively or by judicial decision, one needs a clear notion of what kinds of "processes" would, in a new regime, be considered patentable. The Supreme Court thus far has not been persuaded to extend its interpretation of patentable

---

[306] Newell, *supra* note 15, at 1023. Professor Newell asserts that these conceptual problems do not derive from the nature of algorithms themselves, but only from the efforts to create legal distinctions between "mathematical" and "nonmathematical" algorithms, between "numerical" and "nonnumerical" algorithms, or between "algorithms" and "mental steps," all of which he finds to be "doomed to failure." *Id.* at 1024-25. In this respect, Newell and Chisum seem to be in agreement, for Chisum would repudiate the mental process limitation on patentabiity in general, and not just as it applies to algorithms. *See infra* note 423.

[307] *See infra* notes 410-26 and accompanying text.

[308] Chisum, *supra* note 12, at 1009-19. For the author's discussion of the policy reasons favoring patent protection for algorithms and other program-related inventions, see *infra* notes 459-69 and accompanying text.

[309] For the author's discussion of the policy reasons against patent protection for algorithms and other program-related inventions, see *infra* notes 433-49 and accompanying text. For Professor Newell's concerns about patents for algorithms possibly slowing down the pace of innovation in the programming field, see *infra* note 335 and accompanying text.

[310] For additional discussion of this issue, see *infra* notes 422-26 and accompanying text.

processes beyond the traditional transformation of matter standard.[311] Although Professor Chisum criticizes this restriction as unwarranted,[312] it is fair to ask what alternative definition of patentable subject matter Professor Chisum offers. Because Professor Chisum holds up the CCPA's decision in *In re Musgrave*[313] as "the highwater mark of rationality" concerning what is patentable subject matter,[314] it is appropriate to revisit the *Musgrave* decision.

3. The Musgrave *Decision and Its Standard of Patentability*

Musgrave's claim was for an improved process for seismographic analysis of subsurface geological formations. Some steps in this process were "physical steps" (for example, setting off dynamite to get a seismogram) and some were "mental steps" (measurements and calculations). Only the "mental steps" of the process (some new equations which Musgrave had discovered could be used to reduce the distortions that were produced when seismograms were created and that made the seismograms difficult to interpret) differentiated Musgrave's process from the prior art.[315] Because of this, the Patent Office regarded Musgrave's process as unpatentable subject matter.[316]

---

[311] *See supra* notes 265-68 and accompanying text.

[312] Chisum's article does contain some evidence seemingly contrary to this assertion. For example, he observes that there have been no significant changes in the statutory provision concerning patentable subject matter since the first U.S. patent law of 1793. Chisum, *supra* note 12, at 963. Moreover, he cites case law that endorses the mental process, business method, and printed matter limitations on patentability going back almost as far as a century for each of them. *Id.* at 964-68. Yet, Chisum does not give this history any weight.

Furthermore, he does not give any weight to the long history of decisions that limit the meaning of "process" to those that transform matter. Chisum may be right that Cochrane v. Deener, 94 U.S. 780 (1877), upholding a patent on refining flour, does not directly hold that it is *necessary* for a process to "transform matter" to be patentable. *See* Chisum, *supra* note 12, at 967-68 n.30, 987-88. However, when a case has been cited and quoted as often as *Cochrane* for this proposition, it should be understood to stand for this proposition and should not be lightly swept aside. With the exception of the CCPA's own cases on computer program-related inventions, which do not accept the *Cochrane* limitation, Chisum does not cite any cases that uphold a patent for a process which does not meet *Cochrane*'s transformation standard.

[313] 431 F.2d 882 (C.C.P.A. 1970).

[314] Chisum, *supra* note 12, at 970.

[315] *Musgrave*, 431 F.2d at 885-86.

[316] The Board of Appeals reasoned that if inclusion of a physical step (however obvious or known to the prior art) was enough to make mental steps patentable, "then methods of telling for-

The CCPA majority opinion responded to this argument by saying that the patent statute itself "contains nothing whatever which would either include or exclude claims containing 'mental steps' and whatever law there may be on the subject cannot be attributed to Congress. It is purely a question of case law. That law we, like others, have found to be something of a morass."[317] The CCPA, therefore, repudiated the "mental steps" doctrine and announced a new standard for what was a patentable process:

> We cannot agree with the board that these claims (all of the steps of which can be carried out by the disclosed apparatus) are directed to non-statutory processes merely because some or all the steps therein can also be carried out in or with the aid of the human mind or because it may be necessary for one performing the processes to think. All that is necessary, in our view, to make a sequence of operational steps a statutory "process" within 35 U.S.C. § 101 is that it be in the technological arts so as to be in consonance with the Constitutional purpose to promote the progress of "useful arts."[318]

Because the CCPA perceived Musgrave's process to be in the technological arts, it was patentable.

Although Judge Baldwin concurred in the result in *Musgrave* because the process was to be implemented by machine,[319] he strongly criticized

---

tunes or predicting the activities of the stock market would be patentable providing one included the use of playing cards or a desk calculator in a claim that otherwise is for a nonstatutory algorithm, such as the hypothesized principles underlying human behavior or the fluctuating values of the stock market." *Id.* at 886. *See, e.g.*, "Securities Valuation System," Patent No. 4,566,066.

In discussing one of the claims, the Board noted: "Since these signals are not specified to be electrical, mechanical or optical or to denote any other physical state or a material or thing, the sole connotation here would be that 'signals' (*i.e.*, without a modifier) are synonymous with information or *data*, and are an abstraction and intangible." *Musgrave*, 431 F.2d at 886 (emphasis in original). The Board pointed out that the process also included a "step of exercising human judgment that would be required to interpret these signals to gain any knowledge of the static corrections needed." *Id.* at 887. The Board relied on prior CCPA decisions to support its rejection of Musgrave's process claims as not directed to statutory subject matter. *Id.* at 885-88.

[317] *Musgrave*, 431 F.2d at 890. The CCPA discussed one of the Board's prior decisions to show how slippery a concept "mental steps" is. *Id.* at 892 (discussing *Ex parte* Kahn, 124 U.S.P.Q. (BNA) 511 (PTO Bd. App. 1959)).

[318] *Musgrave*, 431 F.2d at 893 (footnote omitted).

[319] For a discussion of this pre-*Benson* patentability standard, see *supra* notes 63-64 and accompanying text.

the majority opinion in *Musgrave*.[320] Describing the patentability standard promulgated in the majority opinion as a "major and radical shift in this area of the law," Judge Baldwin said that it was also "a serious breach with the time-honored judicial practice of resolving important questions of law on a case-by-case basis."[321] Among the problems Judge Baldwin perceived with the majority's new standard in *Musgrave* was in "interpreting the meaning of 'technological arts.' "[322] He questioned the patentability of intellectual processes that could not or were not intended to be carried out by machine, as well as those that might make a contribution to the liberal arts and the technological arts.[323] He concluded that "what the majority has done will only substitute for one set of problems another possibly more complex set" and predicted that the law would be more confused as a result of the court's apparent enlargement of the bounds of patentable subject matter.[324]

4. Benson's *Clarification of* Musgrave

In *Musgrave*, the CCPA left open the question whether Musgrave's process was patentable because the field of its application was a technological one or merely because it could be carried out by machine. The CCPA seemed to answer this question in *Benson* in favor of the latter interpretation.[325] In rejecting the Patent Office's argument that processes like Benson's were unpatentable because they could be performed mentally (even though, with the advent of computers, they could now be performed by machine), the CCPA drew an analogy to make its point: The

---

[320] *Musgrave*, 431 F.2d at 893-96 (Baldwin, J., concurring).

[321] *Id.* at 893-94. Judge Baldwin noted: "No limitations are placed upon this holding. In effect it is a pronouncement of new law." *Id.* at 894.

[322] *Id.* at 895. Judge Baldwin thought this sounded broader than the "industrial technology" standard that the CCPA had used before. *Id.*

[323] *Id.* at 895-96. For a discussion of the potential patentability of such processes, see *infra* notes 401-08 and accompanying text. Though the *Musgrave* majority opinion indicates that a process involving a "subjective judgment" cannot be patented, Judge Baldwin said that it did not "require much imagination to see the many problems sure to be involved in trying to decide whether a step requiring certain human judgment evaluations is definite or not." *Musgrave*, 431 F.2d at 896.

[324] *Musgrave*, 431 F.2d at 896. For Professor Newell's similar prediction, see *supra* note 306 and accompanying text.

[325] *In re* Benson, 441 F.2d 682 (C.C.P.A. 1971), *reversed sub nom.* Gottschalk v. Benson, 409 U.S. 63 (1972); *see supra* notes 86-88 and accompanying text. But for discussion of more limiting interpretations of the CCPA *Benson* decision, see *infra* notes 398-408.

mere fact that cash registers work with numbers does not keep them from being patentable.[326]

The patentability of a cash register as a machine, however, is not an apt analogy. The Patent Office did not argue that computers or cash registers would be unpatentable as machines merely because they could be used to add or subtract numbers. The Office argued that if the *process* that both cash registers and computers could carry out — addition or subtraction, for example — could also be carried out in a person's head, *that* process would be unpatentable as a mental process.[327] The CCPA's *Benson* decision thus seems to say that a new method of performing addition or subtraction, because it can be carried out on a cash register machine or a computer, is technological enough to be a patentable process even if the patent claims are not limited to machine implementations.[328]

Indeed, Professor Newell, in his article in response to Professor Chisum, uses addition as an example of an algorithm that Chisum's analysis would transform into patentable subject matter.[329] He finds this prospect troublesome for a number of reasons.[330] One is his concern with the possibility that one may infringe a patent merely through the act of thinking.[331] Another is the theoretical nature of mathematical innovations.[332]

---

[326] *Benson*, 441 F.2d at 687.

[327] *See supra* notes 79-80 and accompanying text.

[328] Recall that one of Benson's claims was not limited to machine implementations. *See supra* note 73 and accompanying text. Even the dissenters in *Flook*, who would have found Flook's equation for use in a catalytic conversion process to be patentable, gave multiplication as an example of an unpatentable mental process. *See* Parker v. Flook, 437 U.S. 584, 598 (1978) (Stewart, J., dissenting).

One potential problem with the patentability of something like an algorithm for addition or multiplication is the breadth of the claims that might be made for it since the method could be employed in numerous applications. See *supra* note 100 concerning "abstractness" and "breadth" as problems the Court found with the Benson claims.

[329] Newell, *supra* note 15, at 1026-28. It may be worth noting here that Professor Newell did not read either the CCPA's *Musgrave* or *Benson* decisions; hence, he did not choose addition as an example in order to counterpoint those cases as a lawyer might have done. He simply saw the logical outcome of Chisum's reasoning.

[330] "Doing addition is accomplished by carrying out an algorithm. If algorithms are patentable, then I can keep you from doing addition with the algorithms invented for it. There would be ever so many things that the poor would not be able to do, such as add up their grocery bill." *Id.* at 1027.

That algorithms such as addition, subtraction, and the like have already been invented so we don't have to worry about patents for them is no answer to the analytic puzzle presented by the question of whether these algorithms are patentable. If Newell is right, we may be on the verge of mathematical inventions that are as profound and universal in application as addition. *Id.* at 1028.

[331] Professor Newell also observes that "[a]n identity between algorithms and mental steps leads

Patent law, he observes, generally posits a gap between a basic scientific discovery and the discovery of particular applications of it to some useful end: "The discovery of a natural law or mathematical truth does not wear its practical application on its sleeve, so to speak."[333] However, in computer science, this gap does not exist in the way it exists in other fields. "*All* of computer science is directly related to use. There is essentially no gap, no matter how pure or basic the science is."[334] Newell fears that continued progress in the field of computer science and programming might be jeopardized if patent protection were extended to reach this field's algorithmic advances.[335]

---

to such questions as whether you can keep people from thinking patented thoughts," a position which he regards as "untenable." Newell, *supra* note 15, at 1025; *see also In re* Bernhart, 417 F.2d 1395 (C.C.P.A. 1969) (discussing concern with "mental infringement"); *supra* note 60.

While his reading of Professor Chisum's article suggests that perhaps a "fair use" argument could be made for thinking through a patented process, Professor Newell thinks that it would be difficult to sustain this argument because "[w]e are talking about people who engage in those patented thoughts daily and hourly . . . in the pursuit of their business and who make their money and their livelihood by so doing." Newell, *supra* note 15, at 1025; *see* Eisenberg, *Patents and the Progress of Science: Exclusive Rights and Experimental Use*, 56 U. CHI. L. REV. 1017 (1989).

[332] Recall that the Supreme Court in its decisions on program-related patentability issues was concerned with this as well.

[333] Newell, *supra* note 15, at 1026.

[334] *Id*; *see* Reichman, *Computer Programs as Applied Scientific Know-how: Implications of Copyright Protection for Commercialized University Research*, 42 VAND. L. REV. 639 (1989) (discussing the limits of copyright protection for computer programs because of the scientific character of program innovations). A similar problem is arising in the field of genetic engineering, where theoretical advances and practical applications often go hand in hand. *See* Eisenberg, *supra* note 331.

[335] Newell's article discusses an alternative to the conventional patent model which assumes a net benefit from patenting:

> Consider an alternative model, in which inventions produce, not consumables, but "inventables." That is, suppose the primary effect of every product is to enable additional inventions. . . . New invention comes from the products produced by the original invention, not from the original invention itself. The price increase due to the patent monopoly will restrict the amount of product sold, just as in the original model. But now this implies that the number of new inventions that occur will also be restricted . . . . The patent system could discourage invention rather than encourage it.
>
> This may seem like an extreme alternative model. However, some computer communities may approximate it. There, people take each other's programs freely, then enhance them, and then pass them on to others, who do more of the same. Of course, they also use them as well. But consuming the behavior of a program does not consume the program. Furthermore, it is the possession of the previously invented program that permits the new invention to occur. For the new inventor adds, modifies, enhances, and reshapes the existing system. . . . The capabilities of the system evolve and grow. The motivations for such enhancements are partly that one benefits from the inventions themselves, for one gets to use the enhanced system. But the motivations are also partly those that keep the artist and

Professor Chisum is not entirely clear about his view on the patentability of mathematical innovations. While it is clear that he would approve of a patent for Benson's method, his argument is partly grounded on his view that the Benson method was not "truly mathematical" in nature.[336] It is only by way of a rhetorical question that Chisum seems to endorse the view that mathematical innovations with "direct industrial applications" ought to be patentable.[337] Even then, he does not say what he means by "direct industrial applications."[338]

However, Professor Chisum's main argument in favor of the patentability of computer program algorithms like the Benson algorithm rests on the fact that, like other patentable processes, program algorithms define a

---

> the mathematician creating — it becomes a medium of expression and a coin of the realm. If patentability implies that mostly what is used is left untouched and unenhanced, then the total improvements in the community's software may well decrease, even though some people are induced to work harder at innovating to capture the rewards from patents. They must do their inventing from a poorer base.

Newell, *supra* note 15, at 1033-34. While Newell does not directly argue for this model, he offers it to point out that "the world of algorithms and computers may have a different character than the standard economic model of incentives that underlies the patent law." *Id.* at 1034. This is one of the reasons Newell argues that the traditional patent model may be "broken" when applied to programs and algorithms. *See* Reichman, *supra* note 334, at 652-53 (discussing programs as incremental innovation).

[336] Chisum asks, "But is the [Benson] method one for solving a 'mathematical problem' as the Court states? It would seem that a conversion of decimal numbers . . . is a 'mathematical problem' only in a very loose sense." Chisum, *supra* note 12, at 977; *see also id.* at 961 n.2 (distinguishing between sorting algorithms and "true mathematical algorithms"). Professor Chisum also contrasts the Benson algorithm with the Euclidean algorithm for finding the greatest common divisor of two numbers and describes the latter as "truly mathematical" in nature. *Id.* at 976-77.

[337] Chisum poses this question: "[W]hy are new and useful developments in *mathematics* with direct industrial applications per se excluded from the patent system when developments in all other areas of applied technological knowledge are included?" *Id.* at 1007 (emphasis in original). Although this question suggests that Professor Chisum thinks that mathematical innovations with industrial applications should be patented, he does not quite say what he would do about all of the "non-industrial" or "indirectly" industrial applications of mathematical innovations. See *supra* note 283 concerning AT&T's claim for industrial applications of the Karmarkar algorithm.

[338] In view of the fact that Chisum's question seems to interject "direct industrial applications" as a qualifier on the reach of a patent for mathematical innovations, it is worth asking what the phrase "industrial applications" includes and does not include. Chisum is discussing the *Meyer* and *Pardo* cases when he poses the above rhetorical question; thus, it is fair to use the claims in those cases to ask what "industrial applications" means. *See* Chisum, *supra* note 12, at 1006-09. Pardo's assignee now is seeking to enforce the patent against developers of spreadsheet programs, *see supra* note 249, claiming that it covers the natural order recalculation functions which spreadsheets use. The Meyer invention was for an expert system for assisting medical diagnoses. *See infra* notes 358-73 and accompanying text. Are these "direct industrial applications" or not?

sequence of operations that accomplishes a useful result.[339] From this, he argues that the post-*Benson* case law distinction between "mathematical" and "nonmathematical" algorithms is artificial.[340]

While Chisum's argument has some appeal, it ignores important distinctions traditionally made by patent law. Some algorithms, like those for making steel, have their primary instantiation in the processing of raw materials in a steel plant. Computer program algorithms, such as Benson's method for converting binary coded decimals to pure binary form, have their primary instantiation in computer programs written to implement them. Still other algorithms, such as ones for performing statistical analyses on data, for analyzing the sentence structure in Balzac's novels, or for ordering names and telephone numbers, may have their primary instantiation in written texts. Patentability has traditionally been judged by the nature of the primary instantiation anticipated by the procedure.[341] Instantiation in the text of written works has traditionally been regarded as outside the bounds of the patent system.[342]

Even Professor Chisum himself has written that "the general purpose of the statutory classes of subject matter is to *limit* patent protection to the field of applied technology, what the U.S. Constitution calls the 'useful arts.' "[343] However "practical and useful" they may be, Chisum says that "discoveries . . . in nontechnological arts, such as the liberal arts, the social sciences, theoretical mathematics, and business and management methodology" are not patentable.[344] Mathematics has traditionally been considered part of the "liberal arts."[345]

---

[339] Chisum, *supra* note 12, at 974-77.

[340] *Id.* at 976-77. *But see supra* note 336 and accompanying text (demonstrating that Chisum himself sometimes makes this distinction). For Professor Newell's rejection of this distinction, *see infra* notes 389-94 and accompanying text.

[341] The invention in MacKay Radio & Tel. Co. v. Radio Corp. of Am., 306 U.S. 86, *reh'g denied*, 306 U.S. 618 (1939), for example, was patentable because it was instantiated (*i.e.*, the invention was embodied) in the physical structure of a radio antenna. *See supra* note 232.

[342] For discussion of the "printed matter" limitation on patentability, see *supra* note 36 and accompanying text. For further discussion of the patentability of works instantiated in writings, see *infra* notes 402-08 and accompanying text.

[343] 1 D. CHISUM, *supra* note 2, § 1.01 (emphasis added).

[344] *Id.*

[345] *See supra* note 97 and accompanying text.

VI. The Patentability of Expert Systems and Other Data Processing Programs

Because the post-*Benson* case law tends to make the patentability of program-related inventions turn on the presence or absence of "mathematical algorithms," the underlying similarity between the patentability problems presented by claims for mathematical formulae and those for other information processing methods has generally escaped notice.[346] This section demonstrates that these two apparently different issues are simply two subsets of the same problem.

This section focuses its analysis on the patentability of information processing methods by discussing one example, expert systems. The main reason for discussing expert systems is that the last of the CCPA's program-related patentability decisions, *In re Meyer*,[347] involved claims for such a program. In *Meyer*, ironically enough, the CCPA finally affirmed rejection of patent claims for a program-related process because it mimed the mental steps a person would follow doing the same task.[348] Thus, in *Meyer*, the CCPA brought the computer program patentability dispute back full circle to where it began so many years ago.[349]

In the meantime, however, the Patent Office seems to have had a change of heart on this subject, for, in recent years, it has issued a number of patents for expert systems innovations[350] as well as for a wide variety of

---

[346] *See, e.g., In re* Toma, 575 F.2d 872 (C.C.P.A. 1978) (finding claims for algorithm for natural language translation process to be patentable because they were not for a "mathematical algorithm").

[347] 688 F.2d 789 (C.C.P.A. 1982). Since Meyer's invention was not described by the CCPA as involving an "expert system," the case does not directly address the question of the patentability of expert systems. Nevertheless, the invention was an expert system. *See, e.g.*, E. Feigenbaum & P. McCorduck, The Fifth Generation 65 (1983) (discussing the Cadeseus expert system).

[348] *See infra* notes 366-68 and accompanying text.

[349] *See supra* notes 27-37 and accompanying text.

[350] When the author entered "expert system" into the utility patent part of the LEXPAT service on August 16, 1990, she was informed that there were 226 entries to be searched. Among the first 10 entries, she found the following patents: Patent No. 4,947,095, for an expert system for machine tool equipment; Patent No. 4,945,476, for an "interactive system and method for creating and editing a knowledge base for use as a computerized aid to the cognitive process of diagnosis"; Patent No. 4,943,933, for a "method for translating a data base to knowledge, and apparatus therefor"; Patent No. 4,943,932, for an "architecture for composing computational modules uniformly across diverse developmental frameworks"; and Patent No. 4,942,527, for a computerized management system.

Eleven months earlier, on September 11, 1989, she ran the same search and found 107 entries to be

other data processing inventions.[351] Yet, there are some signs that the Office still regards the "mental process" issue as a serious one because it recently found claims quite similar to Meyer's to be unpatentable,[352] and the Federal Circuit affirmed this rejection in its 1989 decision, *In re Grams*.[353]

After a brief introduction to expert system programs, this part will review the *Meyer* and *Grams* cases. It then turns to the *Merrill Lynch* decision, in which a federal district court upheld the patentability of a computerized brokerage service.[354] *Merrill Lynch*, which was decided between the *Meyer* and *Grams* cases, is inconsistent with those decisions because it

---

searched. A review of the first 10 of these entries yielded the following patents, among others: Patent No. 4,853,873, for a "knowledge information processing system and method"; Patent No. 4,853,850, for a "vehicle computer diagnostic interface apparatus"; Patent No. 4,853,175, for a "power plant interactive display; Patent No. 4,852,173, for design and construction of a binary tree system for language modelling"; and Patent No. 4,847,784, for a "knowledge based tutor."

[351] *See, e.g.*, Patent No. 4,931,928, for a system for analyzing source code; Patent No. 4,905,163, for an "intelligent optical navigator dynamic information presentation and navigation system"; Patent No. 4,839,853, for computerized "information retrieval using latent semantic structure"; Patent No. 4,554,632, for a "method and apparatus for determining if a digital value lies within a range"; Patent No. 4,554,631, for a "keyword search automatic limiting method"; Patent No. 4,559,598, for a "method of creating text using a computer"; Patent No. 4,559,612, for a "sorting device for data words"; Patent No. 4,566,065, for a "computer-aided stenographic system"; Patent No. 4,566,078, for "concurrent multilingual use in data processing system"; Patent No. 4,571,679, for a "fault tree data array"; *see also* Comment, *The Patenting of MIS Computer Programs: One Step Beyond*, 21 PAC. L.J. 761, 779-80, 796 (1990).

It is worth noting that the European Patent Office has rejected program-related claims that involve information processing, sometimes on the ground that they mime human mental processes or do not have the requisite "technical character." *See, e.g.*, summary of EPO rejection of IBM claims for method of automatically detecting and correcting contextual homophone errors in a text document, 12 EUR. INTELL. PROP. REV. D-102-03 (1990); summary of EPO rejection of IBM claims for system for automatically generating a list of expressions semantically related to an input linguistic expression, 12 EUR. INTELL. PROP. REV. D-15 (1990); and summary of EPO rejection of IBM claims for system for automatically abstracting a document, 12 EUR. INTELL. PROP. REV. D-58-59 (1990).

[352] *See In re* Grams, No. 88-1391 (PTO Bd. App. 1988). The PTO Report is somewhat vague about the extent to which the Patent Office still considers the "mental process" doctrine to be a viable limitation on the patentability of program-related inventions. *PTO Report*, *supra* note 2. Toward the very end of the Report, in a section entitled "The CCPA Has Held That Computer Processes Are Statutory Unless They Fall Within a Judicially Determined Exemption," it says, "Arguably, other exceptions such as 'methods of doing business' or 'mental steps' may be raised if a claim is not a true computer process but merely recites that an otherwise nonstatutory process is performed on a computer." *PTO Report, supra note 2, at 570.*

[353] 888 F.2d 835 (Fed. Cir. 1989).

[354] Paine, Webber, Jackson & Curtis, Inc. v. Merrill Lynch, Pierce, Fenner & Smith, Inc., 564 F. Supp. 1358 (D. Del. 1983).

holds that program-related claims are patentable as long as they can be carried out by a machine; *Meyer* and *Grams* indicate that this is an incorrect standard.

A. *Expert System Programs*

An expert system is a computer program or set of programs containing a body of knowledge about a particular domain and a set of rules for the application of this knowledge to specific problems.[355] Expert systems are designed to enable users to call upon them much as the users might call on a human expert in order to get assistance in solving problems in that domain. Among the domains for which expert system programs have already been created are medical diagnoses, mathematics, meteorology, law, investment analysis, agriculture, and chemistry.[356]

The process of building an expert system is often referred to as "knowledge engineering." It typically involves collaboration between someone with a background in computer science and artificial intelligence (the "knowledge engineer") and one or more human experts in the field (the "domain expert"). From the domain expert, the knowledge engineer gains an understanding of the domain knowledge that an expert would need to make an intelligent judgment to solve certain problems in the domain area. The knowledge engineer also gains from the expert an understanding of the procedures, strategies, and rules of thumb ("heuristics") used by the domain expert to solve problems in that field.[357]

From this body of data, the knowledge engineer develops a plan for structuring domain knowledge in a data base and a set of rules (often in "if-then" form) which represent the heuristics that human experts would use to apply that knowledge to solve domain problems.[358] Both of these

---

[355] *See generally* E. CHARNIAK, C. RIESBACK & D. MCDERMOTT, ARTIFICIAL INTELLIGENCE PROGRAMMING (1980); F. HAYES-ROTH, D. WATERMAN & D. LENAT, BUILDING EXPERT SYSTEMS (1983); D. WATERMAN, A GUIDE TO EXPERT SYSTEMS (1986).

[356] Chapter 25 of Waterman's book on expert systems contains summaries about particular expert systems grouped by area of domain. D. WATERMAN, *supra* note 355, at 244-99. Chapter 26 contains a bibliography on expert systems for these domains. *Id.* at 300-35.

[357] *Id.* at 5-10.

[358] *Id.* at 16-23. Chapter 15 of the Waterman book gives an example of the expert system building process, using the insurance business as a domain. This expert system was aimed at improving the ability of insurance adjusters to make more accurate assessments of claims submitted, so as to reduce

will eventually become part of the expert system program. The "inference engine" of an expert system program is that part of the program that contains general problem-solving knowledge. The "interpreter" is the portion used for deciding how to apply domain-specific knowledge to a problem, and the "scheduler" is the part used for deciding when and in what order different pieces of domain knowledge should be applied.[359] Specialized programming languages have been created to aid in the creation of expert system programs.[360]

Expert system programs are generally not intended to displace completely the domain expert whose brain has been "picked" by the knowledge engineer, primarily because expert systems lack robustness at domain boundaries. Nevertheless, these programs can be valuable aids to those who are responsible for solving problems in the domain area and are often characterized as "assistants," "advisors," or "associates" by their developers.[361] Expert systems have several advantages over exclusive reliance on human experts: they do not die; their memories do not fade; they are easily reproduced and distributed; they are low in cost after the initial devel-

---

the products liability exposure for such a business. *Id.* at 162-75.

It is worth emphasizing that the knowledge engineer's method of representing the domain knowledge and the rules derived from the human experts will significantly differ from the way in which the domain expert would represent these things, for the knowledge engineer needs to structure them in a manner which makes it possible to write an effective computer program, that is, make them computable. For a further discussion of the computability of such systems, see *infra* notes 391-94 and accompanying text.

[359] D. WATERMAN, *supra* note 355, at 18-23. This is a good point at which to mention another problem that Professor Newell raises in questioning the patent system's conceptual models for dealing with program-related inventions. While the term "algorithm" is often defined as a sequence of steps to accomplish a particular task, Professor Newell observes that "[c]omputer science takes an algorithm to be any specification that *determines* the behavior of a system. These specifications can be of any kind whatsoever as long as they actually provide the determination through the interpreter. Consequently, the form of the specification need no longer be procedural." Newell, *supra* note 15, at 1032.

Newell gives expert systems as an example of this nonprocedural kind of algorithm:

> The form of many expert systems is simply a collection of if-then rules that provide the knowledge that is needed to perform a task. There is no easy way of seeing such an expert system as a sequence of steps — except that the rule interpreter determines at each moment one rule to fire. . . . If all that counts is the knowledge, then the specifications can look declarative or procedural or any other way.

Id. This would certainly seem to present challenges for claim drafting practices.

[360] D. WATERMAN, *supra* note 355, at 80-83.

[361] *See, e.g.*, Rouse, Geddes & Hammer, *Computer-aided Fighter Pilots*, IEEE SPECTRUM, March 1990, at 38 (discussing the "Pilot's Associate" project). Notice the anthropomorphic quality of the characterization.

opment; and they produce more consistent results over time (not periodically distracted by stress or emotional problems).[362]

Although expert system programs, like all other computer programs, are executed by machine, the creativity they embody generally resides in the representation, organization, analysis, and presentation of data. Thus, expert system programs raise interesting patentability questions.

B. Meyer's *Rejection of Expert System Patent Claims*

The *Meyer* case was the first program-related patentability case to involve an expert system program. Meyer's expert system, known as Cadeseus, is useful for diagnosis of neurological problems.[363] Meyer made both process and apparatus claims for this system.

Some of Meyer's claims recited "algorithms" although these algorithms did not call for numerical calculations.[364] The Patent Office examiner rejected all claims as nonstatutory subject matter under *Benson*.[365] In arguing the case before the CCPA, the Patent Office solicitor characterized the invention:

> as a "diagnostic" or "memory" aid for a physician and emphasized that the invention does not conduct a diagnosis in and of itself, but is used by a doctor when performing a diagnosis to store and to accumulate test responses obtained by this standard process of elimination and to narrow the neurological area of possible malfunction.[366]

---

[362] D. WATERMAN, *supra* note 355, at 12-13.

[363] For a description of the invention, see *Meyer*, 688 F.2d at 790-93. An attending doctor can use Cadeseus to enter data related to patient symptoms and/or results of medical tests. If more information is needed to answer a diagnostic inquiry, the program will respond with a request for specific additional data. If all needed data are at hand, Cadeseus will suggest possible maladies with which the patient might be afflicted, or it might suggest additional tests that could be carried out to validate a diagnosis.

[364] Although Meyer's brief apparently contained an equation, *see Meyer*, 688 F.2d at 792, the claims shown in the opinion made no reference to equations. *Id.* at 792-93.

[365] The examiner regarded the claims to be for a mathematical algorithm. In affirming the examiner's conclusion on this point, the Board of Appeals said: " 'The claims are drawn to a technique of statistical analysis. Data is accumulated from a series of test operations and conclusions are drawn in accordance with a mathematical algorithm.' " *Id.* at 793. The Board emphasized that the steps in the process involved data gathering and manipulation and concluded that the process recited was for a mathematical algorithm. *Id.* at 793-94.

[366] *Id.* at 793. The solicitor also pointed out that the more experienced the doctor and the better

The CCPA's opinion in *Meyer* begins with the standard recitation on the unpatentability of "mathematical algorithms." The CCPA noted that this exclusion from patentability "is consistent with the Court's long-standing exclusion . . . of scientific principles, laws of nature, ideas, and mental processes."[367] The opinion continues by saying that some scientific principles can be expressed in a mathematical format: "However, some mathematical algorithms and formulae do not represent scientific principles or laws of nature; they represent ideas or mental processes and are simply logical vehicles for communicating possible solutions to complex problems."[368] In this part of the *Meyer* opinion, the concept of "mathematical algorithms" seems to become a subset of the "mental process" limitation on patentability, rather than, as some previous CCPA decisions seemed to say, the only category of exclusion from patentability relevant to program-related inventions.[369]

Not surprisingly, the CCPA still cast its announcement of the unpatentability of Meyer's invention in familiar terms by saying that the Meyer claims were for "a mathematical algorithm representing a mental process that has *not* been applied to physical elements or process steps"[370] and were therefore for unpatentable subject matter.

Despite the opinion's emphasis on "mathematical algorithms," it is clear from a close reading that the CCPA would not accept a patent for an innovation which was "concerned with replacing, in part, the thinking processes of a neurologist with a computer."[371] The CCPA noted that Meyer's counsel conceded that this invention "represents a mental process that a neurologist should follow."[372] Because this mental process was not

---

his or her memory, the less the doctor's need for this invention. *Id.*

[367] *Id.* at 794.

[368] *Id.* at 794-95. The presence of a mathematical formula in a claim was merely an indication that there might be a subject matter problem with the claim, but was not itself determinative: "In considering a claim for compliance with 35 U.S.C. § 101, it must be determined whether a scientific principle, law of nature, idea, or mental process, which may be represented by a mathematical algorithm, is included in the subject matter of the claim." *Id.* at 795.

[369] *See, e.g., In re* Toma, 575 F.2d 872 (C.C.P.A. 1978); *In re* Freeman, 573 F.2d 1237 (C.C.P.A. 1978); *see also supra* notes 178-88 and accompanying text.

[370] *Meyer*, 688 F.2d at 796 (emphasis in original). The court made no distinction between the process and apparatus claims. *See id.* at 795 n.3.

[371] *Id.* at 795.

[372] *Id.* Actually, an expert system will organize information about a problem and rules for how the information should be utilized in a different manner than a doctor would who was asked to

"applied to physical elements or process steps in an otherwise statutory process, machine, manufacture, or composition of matter,"[373] the CCPA concluded that it did not satisfy the requirements of section 101. The recent Federal Circuit decision in *In re Grams*[374] involved an invention very similar to Meyer's,[375] and, relying on *Meyer*, the Federal Circuit upheld rejection of the claims as being for unpatentable subject matter.[376]

While the courts in *Meyer* and *Grams* said that they were rejecting the claims because of the "mathematical algorithms" involved, their use of this term was quite different from that of the CCPA in *In re Toma*[377] and

---

articulate his thinking on solving the same problem. See *supra* notes 355-59 and accompanying text concerning the knowledge engineering that goes into the creation of an expert system.

[373] *Meyer*, 688 F.2d at 795.

[374] 888 F.2d 835 (Fed. Cir. 1989).

[375] Grams' claims are discussed at 888 F.2d 836-37. Claim 1 was for a method of diagnosing an abnormal chemical or biological condition in a human being. It consisted of steps which included conducting clinical laboratory tests on the individual, taking values derived from these tests, comparing them with values associated with the parameters in normal individuals, and if the individual's values showed the presence of an abnormal condition, successively testing various parameters to determine the cause of the abnormality. Grams claimed that his method could be used to detect abnormalities in any complex system, including electrical, biological, mechanical, chemical, or combinations thereof. *Id.* at 836. Claim 16 called for the diagnostic method of the other claims to be performed on a programmed digital computer. *Id.* at 841.

From the description given in the opinion, it appears that Grams was seeking a patent for a relatively conventional diagnostic methodology. The claims were not rejected, however, for lack of novelty or for overbreadth, but only because they were for nonstatutory subject matter. The Federal Circuit somewhat acidly noted that the specification "does not bulge with disclosure" on the tests to be performed as part of the method. *Id.* at 840.

[376] The examiner rejected Grams' claims on the grounds that they were for both a mathematical algorithm and an unpatentable business method. *Id.* at 836. The Federal Circuit found it unnecessary to reach the "business method" basis of the rejection because it agreed with the Patent Office that the claims were for an unpatentable mathematical algorithm. *Id.* at 840-41.

The Federal Circuit noted that Grams did not dispute that claim 1 included a mathematical algorithm. *Id.* at 837. It is worth noting, however, that this claim was not directly for any equation, but it did refer to comparing test values for the individual and values for normal individuals. This is "mathematical" in the larger sense of the word, *see infra* notes 389-94 and accompanying text, but not "mathematical" in the sense in which the CCPA generally used the term.

The only "physical process step" claimed by Grams was performance of clinical laboratory tests. The Federal Circuit concluded that this did not make the process statutory because it "merely provides data for the algorithm." *Grams*, 888 F.2d at 840. (Notice that the Federal Circuit used *Abrams*-like language in discriminating between "mental" and "physical" steps in the process.) Likening the claims to those in *Meyer*, the Federal Circuit concluded that a similar result should be reached in *Grams*. *Id.* at 840-41.

[377] 575 F.2d 872 (C.C.P.A. 1978). Neither *Toma* (algorithm for computerized natural language translation process) nor any of the other CCPA data processing cases were ever mentioned in the *Meyer* or *Grams* decisions. *See, e.g.*, Toma, 575 F.2d at 872 (discussed *infra* notes 402-05 and accom-

kindred cases. Because of this, these decisions would seem to give new life to the "mental process" ground for rejecting program-related patents. This is especially clear in *Meyer* where the CCPA emphasized the invention as one aimed at simulating human thinking. In these two cases, the appellate judges finally grasped something essential about digital computers and computer programs that had long eluded them.

C. *The Merrill Lynch Patent on Computerized Brokerage Services*

One finds a very different analysis of the patentability of an information processing invention in the trial court decision upholding the validity of an apparatus patent issued to Merrill Lynch's assignor for the "cash management account" (CMA) system that Merrill Lynch had computerized and offered as a service to its customers.[378] This system allowed information about different accounts that customers had with the firm to be processed so that money could be transferred among accounts as needed.[379] Paine, Webber, which was using the same combination of services as the CMA system implemented by a different computer program, argued that Merrill Lynch's patent was for an unpatentable computer program algorithm and for an unpatentable "business method."

---

panying text); *In re* Phillips, 608 F.2d 879 (C.C.P.A. 1979) (discussed *supra* notes 237-39 and accompanying text).

[378] Paine, Webber, Jackson & Curtis, Inc. v. Merrill Lynch, Pierce, Fenner & Smith, Inc., 564 F. Supp. 1358 (D. Del. 1983). Paine, Webber sought a a declaratory judgment that the patent was invalid; Merrill Lynch counterclaimed for patent infringement. Paine, Webber moved for summary judgment on the invalidity of the patent on subject matter grounds. *Id.* at 1360-61. The court denied this motion. *Id.* at 1369.

The *Merrill Lynch* decision was the subject of a thoughtful analysis in Note, *supra* note 25, which concludes that the district court erred in upholding the Merrill Lynch patent. *But see* Meyer, *Patentability of Business Methods Implemented by Computer*, COMPUTER LAW., Feb. 1985, at 12 (arguing that the court correctly upheld the patent).

Litigation similar to that in *Merrill Lynch* was initiated by College Savings Bank, owner of patents for a "method and apparatus for funding future liability of uncertain costs," Patent No. 4,722,055 and Patent No. 4,752,877, against Centrust Savings Bank for offering its customers a similar certificate of deposit account earning variable rates of interest based on an index of college costs as that which had been patented. College Savings Bank v. Centrust Bank, No. 89-824 (S.D. Fla. filed May 11, 1989); *see also* claims in *In re* Johnston, 502 F.2d 765 (C.C.P.A. 1974), *rev'd sub nom.* Dann v. Johnston, 425 U.S. 219 (1976) (discussed *supra* notes 155-60 and accompanying text.)

[379] The CMA system is described in *Merrill Lynch*, 564 F. Supp. at 1361-62. Claims 1 and 3 of the patent are recited *id.* at 1364-65.

Following the standard post-*Benson* CCPA analysis, the court looked at the claims to see if they recited a mathematical formula. Not finding one, the court decided that the patent did not claim an algorithm in the *Benson* sense;[380] hence, *Benson* and its progeny did not require the striking down of this patent. (No mention is made of the *Meyer* decision, although it was the CCPA's most recent ruling on the patentability of an information processing method.)

The "business method" attack on the Merrill Lynch patent seemed to be somewhat more difficult for the court to resolve.[381] The court agreed that the Merrill Lynch patent "effectuate[d] a highly useful business method and would be unpatentable if done by hand."[382] However, the court decided that this was irrelevant because the CCPA "ha[d] made clear that if no *Benson* algorithm exists, the product of a computer program is irrelevant, and the focus of analysis should be on the operation of the program in the computer."[383] The court found the claims to be for statutory subject matter "because the claims allegedly teach a method of operation on a computer to effectuate a business activity."[384] The court

---

[380] *Id.* at 1366-68. The court relied heavily on prior information processing cases to support this part of its ruling. *In re* Pardo, 684 F.2d 912 (C.C.P.A. 1982); *In re* Phillips, 608 F.2d 879 (C.C.P.A. 1979); *In re* Toma, 575 F.2d 872 (C.C.P.A. 1978).

The Supreme Court's *Diehr* decision was mentioned once in a footnote in the *Merrill Lynch* opinion for the proposition that the Supreme Court had endorsed a narrow view of the term "algorithm." *Merrill Lynch*, 564 F. Supp. at 1367 n.6.

[381] The court in the *Merrill Lynch* case did not find the business method issue to be a serious problem. *But see Ex parte* Murray, 9 U.S.P.Q.2d (BNA) 1819 (PTO Bd. App. 1988). The *Murray* court denied a patent for a computerized method of providing banking services on the ground that the application sought a patent on a business method. The *Merrill Lynch* case was said to be distinguishable because it involved apparatus, not method claims. The Board also denied the patent because the claims were for a *Benson*-like algorithm: "[I]nasmuch as we find that appealed claims preempt an algorithm for calculating expenses in the *Benson* sense, and involve no more than gathering data, calculating, and forwarding information to customers via printed matter, the claims are nonstatutory under § 101." *Id.* at 1821.

The PTO Report on the patentability of program-related inventions is equivocal about the implications of the "business method" rule for program innovations. *See supra* note 352; *PTO Report*, *supra* note 2, at 570 (seeming to approve of the *Merrill Lynch* decision).

[382] *Merrill Lynch*, 564 F. Supp. at 1369.

[383] *Id.*

[384] *Id.* The court did not discuss a single "business method" case, nor did it inquire about the reasons that might underlie the rule against patenting business methods. As in its section on algorithms, the court made no mention of the *Meyer* case, which might also have been regarded as a method of conducting an aspect of a medical business, nor did it attempt to explain why "computerizing" the method made it patentable.

likened Paine, Webber's attack on the CMA patent to the Patent Office's challenge to Toma's algorithm for computerized translation of written texts.[385] Because the CCPA had upheld Toma's claims on the ground that the method was to be done by machine, the court upheld the Merrill Lynch patent.[386]

D. *Reflections on the Patentability of Information Processes*

Merrill Lynch and the CCPA's earlier information processing cases cannot be reconciled with the reasoning or the outcome in the *Meyer* and *Grams* cases.[387] All of these cases involved data processing innovations capable of being performed on a digital computer; none of the claims recited equations. *Merrill Lynch* and kindred cases make patentability turn on the presence or absence of equations. Although these opinions recognize that claims for such "mathematical algorithms" are unpatentable under *Benson*, they hold that claims for other kinds of information processing algorithms are patentable because they can be carried out by

---

One argument Merrill Lynch made to overcome the "business methods" problem with its patent was that this doctrine pertained to "process" claims and was inapposite in cases where "machine" claims were being made. *Id.* at 1365. The court said that it was not basing its decision on whether the claims were in method or apparatus form because such labels are not determinative. *Id.* at 1366.

[385] *Id.* at 1369. The patent examiner rejected Toma's claims for a natural language translation method in part because he regarded them as making a contribution to the "liberal arts" rather than to the "technological arts." *See In re* Toma, 575 F.2d 872, 877-78 (C.C.P.A. 1978). However, the CCPA decided that because the method was to be performed by computer, it made the requisite contribution to the "technological arts." *Id.*

The same dispute had erupted earlier in *In re* Johnston, 502 F.2d 765 (C.C.P.A. 1974), *rev'd on other grounds sub nom.* Dann v. Johnston, 425 U.S. 219 (1976), where the Patent Office denied the patentability of claims for a computerized financial record-keeping system, arguing that it made a contribution to something other than the technological arts. The CCPA, relying on the intended mechanical implementation, found a contribution to the technological arts. *See id.* at 771. The court in the *Merrill Lynch* case also made note of the Supreme Court's decision in *Johnston* not to address the patentable subject matter issue as implicitly supporting its ruling. *Merrill Lynch*, 564 F. Supp. at 1369.

[386] For further discussion of *Toma*, see *infra* notes 402-05 and accompanying text. There was no appeal of the *Merrill Lynch* decision because Merrill Lynch's principal competitors decided to take patent licenses instead. *See* R. BROWN & R. DENICOLA, CASES ON COPYRIGHT 125 (4th ed. 1985).

[387] While the Federal Circuit may well try to create some new rationale for distinguishing between the *Meyer-Grams* line of cases and the *Merrill Lynch-Toma* line of cases, it seems inevitable that the conflict between these two lines of cases will come to a head and that some decision will have to be made on the basic issue which underlies them both.

machine and have not been made unpatentable by *Benson*.[388]

Professor Newell asserts that the patent case law distinction between "mathematical" and "nonmathematical" algorithms is indefensible.[389] He and other computer scientists would regard an expert system method, such as Meyer's method, to be as "mathematical" in character as any equation. This assertion may run counter to popular perception, which tends to associate the term "mathematics" with numerical computations, equations, and the like. Computer scientists, however, understand "mathematics" to be a field concerned with defining abstract relationships among concepts and with defining rules about how those concepts should be manipulated.[390]

Professor Newell, commenting on Professor Chisum's article, offers the following explanation as to why no meaningful distinction can be made between algorithms and mental processes:

> The main line of progress in psychology for the last thirty years (called cognitive psychology) has been to describe human behavior as computational. *We model what is going on inside the thinking human brain, as the carrying out of computational steps. Therefore, humans think by means of algorithms. Sequences of mental steps and algorithms are the same thing.* . . . It is not important whether you accept this computational view of human thinking. . . . What is important is that such a view is a major one in the study of the human mind — that many psychologists see the mind in this

---

[388] *See supra* notes 186, 386, and accompanying texts.

[389] *See* Newell, *supra* note 15, at 1024.

[390] *See* J. HAUGELAND, *supra* note 60, at 15-44 (discussing the history of the philosophy of mind and evolving concepts of mathematics and of thinking as computational). Here is Haugeland's synopsis of these developments:

> Galileo "abstracted" Euclid's methods away from purely spatial problems so he could apply them to physics, and now Descartes drives the principle to its radical conclusion: geometry, algebra, and physics are all equally just "applied math." Mathematics *as such* is not concerned with any specific subject matter (figures, numbers, motions, or what have you), but only with the very abstract relationships that might be found in them, or in any other objects. . . .
>
> Though mathematical notations were the model for Descartes's new ideas, he soon extended them to cover everything meaningful — especially thoughts. In other words, he regarded thoughts themselves as symbolic representations, fundamentally analogous to those used in mathematics; and, as a result, he could apply all his conclusions about representations in general to the special case of the mind.

Id. at 31-32 (emphasis in original).

> way and that thousands of technical papers are written from within this view, covering large expanses of psychological phenomena.[391]

Among similar lines, Professor Haugeland states:

> According to a central tradition in Western philosophy, thinking (intellection) essentially *is* rational manipulation of mental symbols (viz., ideas). . . . Computers . . . can manipulate arbitrary "tokens" in any specifiable manner whatever; so apparently we need only arrange for those tokens to be symbols, and the manipulations to be specified as rational, to get a machine that *thinks*. In other words, AI [artificial intelligence] is new and different because computers actually do something very like what minds are supposed to do.[392]

Under this view of intellection and computing, an algorithm for representing how a legal problem can be solved is just as "mathematical" as an algorithm for addition, for finding the lowest common divisor for two numbers, or for conversion of binary coded decimals to pure binary form.[393] For this reason, it is just as amenable to a representation that will make it "computable" as is addition, and it is just as capable of being instantiated in a computer program. Such an algorithm could, of course, also be instantiated in thought or in everyday practice. This means that a patent on the algorithm could be infringed by thinking through its steps or by practicing it in everyday life.[394]

The *Meyer* and *Grams* courts seemed to recognize this possibility implicitly, for the judges in these two cases characterized the claims as mathematical and applied *Benson* to deny their patentability. *Meyer* and *Grams* are indeed consistent with the Supreme Court rulings in computer program cases, and they do indeed call into question the subject matter rulings in the other data processing cases.

While Professor Chisum would favor the *Merrill Lynch* line of cases over *Meyer* and *Grams*,[395] he would argue that they do not go far enough.

---

[391] Newell, *supra* note 15, at 1024-25 (emphasis added).

[392] J. HAUGELAND, *supra* note 60, at 4 (emphasis in original).

[393] Waterman gives a number of examples of transforming domain data and expert heuristics into a set of expert system rules that are "computable," even though they do not involve numerical calculations. *See* D. WATERMAN, *supra* note 355, at 55-57 (prospecting expert system rules involving no numerical computations), 171-75 (expert system rules for evaluating personal injury cases).

[394] *See infra* notes 406-08 and accompanying text.

[395] Professor Chisum asserts that the CCPA's decision in *Meyer* is impossible to square with its

His proposed remedy to the conflicting lines of cases regarding computer program-related inventions is to overrule *Benson*,[396] and he argues that this will mean the end of analytic confusion in the case law.

Let us, then, inquire as to whether Chisum's optimistic predictions are likely to be fulfilled by focusing now on the patentability of information processing methods rather than on equations. Once again, we start with the *Musgrave* standard of patentability which Professor Chisum endorses.[397]

Musgrave left open an issue of considerable significance to the breadth of its ruling: whether the field of application itself has to be technological for the process to be patentable, or whether it is enough to make a process "technological" so that it can be carried out by machine.[398] While the CCPA's *Benson* decision can reasonably be read to support the latter interpretation,[399] it contains some statements that raise questions as to how broad an interpretation should be given to it. The CCPA observed, for example, that the only practical implementation of the Benson method was in a programmed computer and that the Benson method enhanced the internal operations of a computer.[400] Were these, then, the "true" tests for patentability? Among the other unanswered questions from *Benson* and *Musgrave* is what should be done with claims for an algorithm making a contribution to arts other than the technological arts and having a practical implementation without the use of computers.[401]

---

decision in *In re* Pardo, 684 F.2d 912 (C.C.P.A. 1982), with regard to the "mathematical algorithms" these two cases were said to involve. *See* Chisum, *supra* note 12, at 1006-09. The author agrees.

The discrepancy between these two cases is even more surprising given that Judge Miller wrote both opinions, and the opinions were issued only 41 days apart. While Professor Chisum blames *Benson* for the conceptual problems of these two cases, *id.* at 1009, the author attributes the different outcomes in these two cases to the perceived differing nature of the invention. Pardo's method (described in the opinion as a compiler) may have seemed more conventionally machine-like than Meyer's neurological assistant.

[396] *See supra* notes 297-99 and accompanying text.

[397] *See supra* note 318 and accompanying text.

[398] *See supra* notes 66-68 and accompanying text.

[399] *See supra* notes 82-86 and accompanying text.

[400] *See supra* notes 87-88 and accompanying text.

[401] Although the *Musgrave* patentability standard which Professor Chisum endorses would, at first blush, seem to limit the patentability of algorithms to those which are in the "technological arts," the question which neither Professor Chisum nor the CCPA in its pre-*Benson* decisions answered was what the term "technological arts" really includes. Is an algorithm by its very nature "technological" because it involves a precise set of steps to achieve a result? If so, a graduate student's method of

While the Supreme Court's reversal of the *Benson* ruling deferred consideration of the "practical utilization" issue, the CCPA did have occasion after the Supreme Court's *Benson* decision to address some of these questions about what interpretation to give to the CCPA's *Musgrave* and *Benson* decisions. The patent examiner in *In re Toma* rejected claims for an algorithm for natural language translation, in part because the method did not enhance the internal operations of the computer, and in part because it was "in the liberal arts" rather than the "technological arts."[402] The CCPA decided both that the method was in the technological arts because it could be carried out by machine[403] and that enhancing the internal operations of a computer was not necessary to make the process patentable.[404] Interestingly enough, perhaps because it had begun to recognize that *Musgrave* did not provide any meaningful guidance about what might and might not be a patentable process, the CCPA disclaimed in *Toma* that it had ever intended for its statements in *Musgrave* to be a standard for judging the patentability of processes.[405]

Toma, like *Benson*, is arguably a case in which the only practical implementation of the algorithm was in a computer. That still leaves open the question of what to do with patent claims for a process that could be practically implemented either by computer (in an expert system, for example) or without it (writing out the expert system method in a book, for example). It is certainly consistent with *Musgrave*, *Benson*, and *Toma* to say that such an algorithm would be patentable because it too can be implemented in a computer.

Let us consider an example of this kind. Suppose, for example, that someone applied for a patent on an algorithm for analyzing and resolving certain kinds of problems in contract disputes, and that this algorithm was instantiated in an expert system program and in a book about the al-

---

analyzing the structure of Balzac's novels would seem to be patentable, and easily so, if the method can be carried out by computer. The CCPA's decisions in *Musgrave*, *Benson*, and *Toma* seem to consider that if a process can be carried out on a computer, it is enough to make the process "technological." Now that the range of things which can be processed and displayed by computer is so very broad, the *Musgrave* standard can be seen as not being very discriminating in what it would take to satisfy the "technological arts" requirement.

[402] 575 F.2d 872, 877 (C.C.P.A. 1978).

[403] *Id.* at 874, 877.

[404] *Id.* at 877-78.

[405] *Id.* at 878.

gorithm.[406] If a patent issued on this algorithm, would a judge's use of the algorithm to resolve a contract dispute infringe the patent? Would infringement depend on whether he or she had gotten the algorithm from the book or the expert system? (Even independently deriving the algorithm from his or her analysis of contract case law would, of course, not be a defense, for independent development is never a defense in a patent lawsuit.) Would it infringe the patent for lawyers to use the algorithm as the basis of an argument in their briefs or to advise clients based on its analysis?[407] These may sound like frivolous questions, but they are instead very serious ones, made all the more serious by the fact that not only contract law data, but all manner of data can now be processed in a computer.[408]

Even this short exposition of post-*Musgrave* patentability questions

---

[406] As in *Toma*, this advance might well be thought of as a contribution to other "arts" besides a technological one. To make it clearer that a contribution is being made to more than just the technological arts and that the method might be practically utilized in something other than a computer, I hypothesize a book embodying the method as well. See *supra* note 356 and accompanying text and D. WATERMAN, *supra* note 355, at 267-70, concerning expert systems in law. *See also* A. GARDNER, AN ARTIFICIAL INTELLIGENCE APPROACH TO LEGAL REASONING (1987).

[407] Another question that would arise is whether teaching the algorithm in a contracts course in law school would infringe such a patent. It is not clear that this would be within the set of uses for which Professor Chisum would be willing to permit a fair use defense, *see infra* note 446, for teaching the algorithm would provide law students with a method that they could utilize in commercial practice of law for which they could charge money to clients. Therefore, it would not be a use of the algorithm purely for research purposes.

[408] Recall that *Benson* itself can be characterized as a data representation case. *See supra* notes 94-95 and accompanying text. *Toma* can also be characterized in this manner. After describing Toma's method as a human might perform it, the CCPA stated:

> While it is convenient to describe the steps of the program as if they were being performed by a human translator, in fact, nothing of the kind is happening. Rather, the computer is carrying out a series of unthinking, abstract mathematical operations on the abstract values stored in the memory of the computer. The program functions independently of the meaning or significance of the data on which it is acting. The fact that the program is formed in a high level programming language, which makes the program appear to give significance to the machine operation, does not change the fact that the machine is actually carrying out a series of abstract steps which have nothing to do with translating between natural languages. If a different kind of information were fed into the computer, the program used in this invention could conceivably perform a function totally different from translating.

In re Toma, 575 F.2d 872, 874 (C.C.P.A. 1978); *see also In re* Bradley, 600 F.2d 807, 812 (C.C.P.A. 1979), *aff'd by an equally divided court sub nom.*, Diamond v. Bradley, 450 U.S. 381 (1981) (giving a method of organizing numbers for a telephone book as an example of a nonmathematical data manipulation that might be patentable because it is not unpatentable under *Benson*).

demonstrates that although Chisum may be right that overruling *Benson* would resolve one set of difficult and confusing patentable subject matter problems, the predictions both of Professor Newell, in his article in response to Chisum, and of Judge Baldwin, in his concurrence in *Musgrave*, may be equally accurate. To follow the *Musgrave* path instead of the *Benson* path would be to substitute one set of difficult conceptual problems for an equally hard or harder set.[409]

The problems derive from the essential nature of computer programs. Programs defy the conceptual molds provided by the traditional patent and copyright systems for understanding how to protect intellectual work. The computer scientist and mathematician John von Neumann realized that it would be possible to construct a computer that could store not only the data on which the computer was to operate, but also sets of instructions (computer programs) to guide its operations.[410] By executing different sets of stored instructions, a general purpose digital computer could "become" different machines.[411] The von Neumann machine operates by processing symbolic representations (encoded instructions which prescribe the order in which the computer is to perform specified functions to accomplish a given task).[412] With von Neumann's discovery, machines could become writings, and writings could become machines.

It is no wonder, then, that the existing intellectual property law systems have found it so difficult to respond to the phenomenon of computer programs. Programs are, in truth, too much of a mechanical process to fit comfortably in the copyright system and too much of a writing to fit com-

---

[409] *See supra* notes 305, 324, and accompanying texts.

[410] This meant that a computer did not have to be re-wired for each new task it had to perform. *See, e.g.*, Gemignani, *supra* note 62; Patterson, *Microprogramming*, SCI. AM., March 1983, at 50-52.

[411] Each program represents a new configuration for the internal operations of the general purpose machine. While a particular program is running, it converts the machine to a special purpose machine. Thus, computer scientists refer to general purpose digital computers as "universal machines." *See, e.g.*, A. HODGES, ALAN TURING: THE ENIGMA 293-95, 318-21 (1983). Hodges quotes the mathematician and computer scientist Alan Turing as saying: "We do not need to have an infinity of different machines doing different jobs. A single one will suffice. The engineering problem of producing various machines for various jobs is replaced by the office work of 'programming' the universal machine to do these jobs." *Id.* at 293.

[412] Computers can be accurately defined as " 'device[s] capable of solving problems by accepting data, performing prescribed operations on the data, and supplying the results of these operations.' Hence a computer operates on the information which it gets in order to obtain new information or to transform the information into a more useful form." H. HANNEMAN, *supra* note 37, at 3.

fortably in the patent system.[413] They are a hybrid — both writing and machine at the same time — in a legal system that has generally assumed that an intellectual product is either a writing (and hence copyrightable) or a machine (and hence patentable), but not both at the same time.[414] While computer programs may not be the only hybrid phenomenon ever to have existed,[415] they are a particularly valuable hybrid for which protection is of considerable economic importance both now and for the future.[416]

Rather than directly recognizing the hybrid quality of programs, Congress extended copyright protection to computer programs as a kind of writing.[417] Judges in software copyright cases have often struggled to analyze programs as if they were just like any other "literary work"[418] and

---

[413] *See, e.g.*, OTA BACKGROUND PAPER, *supra* note 3, at 1; *see also Samuelson Testimony*, *supra* note 14. When testifying before Congress concerning software intellectual property law, Mitch Kapor, co-developer of Lotus 1-2-3, referred to software as "crystallized mind stuff." *Oversight Hearing on Computers and Intellectual Property Before the Subcomm. on Courts, Intellectual Property and the Administration of Justice of the House Judiciary Committee*, 101st Cong., 2d Sess. 5 (Mar. 7, 1990) (statement of Mitchell D. Kapor, Chairman and Chief Executive Officer, ON Technology, Inc.) [hereinafter *Kapor Testimony*]. Kapor was wary about patent protection because of the special character of software.

[414] Congress has the power "to promote the Progress of science and the useful Arts, by securing for limited Times to Authors and Inventors the exclusive Right to their respective Writings and Discoveries." U.S. CONST. art. 1, § 8, cl. 8. This clause has generally been understood as a dual authorization: first, Congress is authorized to pass laws securing to "authors" exclusive rights in their "writings" in order to promote "science" (by which the Founding Fathers meant "knowledge"). Congress has so acted by passing copyright statutes. Second, Congress is authorized to pass laws securing to "inventors" exclusive rights in their "discoveries" in order to promote progress in the "useful arts." Congress has so acted by passing patent statutes. *See, e.g.*, Kline, *Requiring an Election of Protection For Patentable/Copyrightable Computer Programs*, 6 COMPUTER/LAW J. 607, 647 nn.185-86 (1986) (citing authorities on this point); *see also* OTA BACKGROUND PAPER, *supra* note 3, at 34. Machines, because they are part of the "useful arts," are protected (if at all) through the patent system, whereas writings are protected through the copyright system.

[415] For an in-depth study of other "hybrids" raising similar problems, see Reichman, *Design Protection and the New Technologies: The United States Experience in a Transnational Perspective*, 19 BALT. L. REV. ____ (forthcoming 1991).

[416] *See* OTA BACKGROUND PAPER, *supra* note 3, at 1 (estimating U.S. revenues for software at about $60 billion for 1988 and discussing concerns about legal protection issues).

[417] Pub. L. No. 96-517, 94 Stat. 3015 (1980) (codified at 17 U.S.C. §§ 101, 117 (1988)). *See* CONTU FINAL REPORT, *supra* note 3, at 1-2, 9-10 (recommending enactment of software copyright amendments and discussing the written character of programs). *But see* Samuelson, *supra* note 51, at 727-749 (arguing that the utilitarian character of computer programs makes them an unsuitable subject matter for copyright).

[418] *See, e.g.*, Whelan Assocs., Inc. v. Jaslow Dental Laboratory, 797 F.2d 1222 (3d Cir. 1986), *cert. denied*, 479 U.S. 1031 (1986); *see also* Samuelson, *Reflections on the State of American*

have been confounded by a host of questions which arise from the technological nature of programs for which copyright law has no ready answer.[419] Patent law, which has traditionally regarded writings and the intellection they embody as outside its domain, has slowly, and with mixed results, moved toward protecting the valuable abstractions embodied in programs. However, just as judges in copyright cases have found it difficult to perceive and respond to the functionality of programs, judges in patent cases have ignored the written and symbolic character of software. Neither copyright nor patent law has been able to come to terms with the essential nature of computer programs, and that is why the law in this area is so confusing.[420]

Lawyers want to be able to make firm distinctions and to make legal rulings turn on these distinctions. However, even seemingly meaningful distinctions, like those between computer programs and computers or between programs and data, have a measure of artificiality to them. There is no fixed dividing line between computer programs and computers because anything that can be implemented in software can also be implemented in

---

Software Copyright Law and the Perils of Teaching It, 13 COLUM./VLA J. LAW & ARTS 61, 62-65 & n.15 (1988) (discussing the dual meaning of "literary work" in copyright law and the schizophrenic manner in which the court used the "literary work" analogy in *Whelan*).

[419] For example, is there a "right" under copyright law to develop a program that has portions of code similar to another program in order to make it compatible with a particular hardware or software system? Different courts have responded differently to this question. *Compare* Apple Computer, Inc. v. Franklin Computer Corp., 714 F.2d 1240 (3d Cir. 1983) (rejecting compatibility as a defense), *cert. dismissed*, 464 U.S. 1033 (1984), *with* NEC Corp. v. Intel Corp., 10 U.S.P.Q.2d (BNA) 1177 (N.D. Cal. 1989) (finding no infringement because similarities were attributable to achieving compatibility). *See generally* Menell, *Tailoring Legal Protection for Computer Programs*, 39 STAN. L. REV. 1329 (1987) [hereinafter Menell, *Tailoring*]; Menell, *An Analysis of the Scope of Copyright Protection for Computer Programs*, 41 STAN. L. REV. 1045 (1989) [hereinafter Menell, *An Analysis*].

[420] If, for example, one focuses on programs as processes directing the operation of computers or on the interchangeability of hardware and software, one will conceive of programs as raising no patentable subject matter problems at all. If, on the other hand, one focuses on programs as copyrightable writings, or on program algorithms as equivalent to mental processes, one will conceive of programs as being unpatentable. In the early Patent Office/CCPA struggles over the patentability of program-related inventions, each of the combatants focused on different aspects of the same phenomenon. Neither was able to see the whole elephant.

Similarly, if one views a computer program as a process for operating a computer, it would seem, under traditional principles of copyright law, to be uncopyrightable subject matter. If, however, one views a computer program as a written text, it seems as copyrightable as any other written text. *See* Samuelson, *supra* note 418 (discussing the varying characterizations of programs as writings and machine processes).

hardware. Professor Newell explains why there is no firm line between data and programs:

> As anyone in computer science knows, the boundary between data and program — that is, what is data and what is procedure — is very fluid. In fact, . . . there is no principled distinction in terms of form or representation of which is which. What counts is the total body of knowledge represented somehow in the assembled symbolic expressions. This totality determines the ultimate behavior of the machine.[421]

To come to terms with the legal implications of this fluidity is no easy task.

The law, however, must eventually come to terms with the true nature of programs and decide what role, if any, patents should play in protecting program innovations. The question is whether this decision should be made by the courts or by the legislature.

Professor Chisum sees no difficulty in having the courts rule in favor of the patentability of computer program algorithms[422] and a wide variety of other innovations.[423] Twice in the past decade, the Supreme Court has arguably opened the door to the broadest possible interpretation of patentable subject matter when it has spoken of Congress as having intended to make patentable " 'anything under the sun that is made by man.' "[424] However, these statements are dicta at most, far broader than the actual holdings in the cases would warrant, and both cases were decided by five-to-four votes. One of these cases was *Diehr*, and as was shown earlier, the *Diehr* Court did not question its earlier *Benson* ruling and expressly re-

---

[421] Newell, *supra* note 15, at 1033.

[422] *See supra* notes 296-303 and accompanying text.

[423] Chisum argues not only that computer program algorithms should be patentable, but that the "mental steps," "printed matter," and "business methods" limitations on patentable subject matter should be repudiated: "All three limitations suffer from a common infirmity — the absence of a firm footing in either statutory language or well-reasoned, extra-statutory policy." Chisum, *supra* note 12, at 964. All three kinds of limitations have arisen in computer program-related patent cases. *See supra* notes 60, 381-86, and accompanying texts. Eliminating these impediments to patentability would, of course, have far broader consequences than simply opening the patent system to the protection of all computer program innovations. Michael Milken could now patent his method for junk bond offerings.

[424] Diamond v. Diehr, 450 U.S. 175, 182 (1981); Diamond v. Chakrabarty, 447 U.S. 303, 309 (1980). Both opinions quote this language from the House and Senate Reports on the 1952 Patent Act. *See* S. REP. No. 1979, 82d Cong., 2d Sess. 5 (1952); H.R. REP. No. 1923, 82d Cong., 2d Sess. 6 (1952).

tained the "transformation" interpretation of what processes are patentable.[425]

This article asserts that there is simply too much at stake — not only for the computer software industry, but for the public at large — for such an important change in the subject matter boundaries of the patent system to be made by the courts, especially given the courts' befuddlement thus far about computer program innovations in patent cases. Perhaps computer program algorithms and other information products and processing methods need the kinds of incentives the patent system would provide. A careful study should be made of this and other issues before deciding to extend patent protection to all manner of program-related innovations.[426]

---

[425] *See supra* notes 265-68 and accompanying text.

[426] *See* Comment, *supra* note 215, at 357 (arguing that neither the Supreme Court nor the CCPA is qualified to decide whether patenting of program-related innovations would further the objectives of the patent law and arguing for the need for congressional action).

Among the issues which should be addressed as part of this inquiry is what patentability standards apply in other nations. It would, of course, take an entire article (or book) to cover adequately the subject of computer program patentability standards for all other nations in the world. *See generally* H. HANNEMAN, *supra* note 37. Many articles discuss international patent standards for program-related inventions. *See, e.g.*, Hart, *Application of Patents to Computer Technology — UK and the EPO Harmonization?*, 11 EUR. INTELL. PROP. REV. 42 (1989); Hoffman, Grossman, Keane & Westby, *Protection for Computer Software: An International Overview: Part 2*, 11 EUR. INTELL. PROP. REV. 7 (1989) [hereinafter *An International Overview*]; Stoltysinski, *supra* note 115; Sumner & Plunkett, *Powerful New Software Protection in Europe: The Patent Trend Continues*, COMPUTER LAW., Oct. 1987, at 1.

While the recent trend in the international community has been to recognize the patentability of some program-related inventions (*see generally* H. HANNEMAN, *supra* note 37; Sumner & Plunkett, *supra*), other nations seem to put far greater emphasis than the U.S. Patent Office on limiting the scope of patentable program innovations to those with a traditionally "industrial character." *See generally* H. HANNEMAN, *supra* note 37. *See also* Hart, *supra*, at 42-43, 45 (discussing the United Kingdom's patent statute requiring "industrial application" and a British case in which a patent application by Merrill Lynch for an automated securities trading system was denied because it did not meet the patentability standards and was essentially a method of doing business); *An International Overview*, *supra*, at 8-11 (discussing developments in France and Germany).

Other countries have stated more clearly than the U.S. Patent Office that not only are computer programs per se unpatentable, but "schemes, rules and methods of performing mental acts, playing games or doing business" are also unpatentable, and that these rules have relevance to the patenting of program-related inventions. *See An International Overview*, *supra*, at 10 (citing a French statute to this effect); *see also* Convention on the Grant of European Patents, Oct. 5, 1973, art. 52, para. 2(c). Other countries have also stated more clearly than the U.S. Patent Office that program-implemented machine improvements may be patentable, but that program improvements alone are not patentable. *See, e.g.*, *An International Overview*, *supra*, at 10 n.104; *see also* Appleton, *European Patent Convention: Article 52 and Computer Programs*, 7 EUR. INTELL. PROP. REV. 279 (1985). Moreover, German courts have ruled that program algorithms and organizational rules for the selection, ar-

Based on such study, Congress can then decide what should be done about patents for program-related inventions.

VII. THE WIDER DEBATE OVER SOFTWARE PATENTS

As reflected in the popular press,[427] the current "controversy" over software patents has nothing whatsoever to do with the fine points of the court decisions, doctrinal analysis, and Patent Office practice on which this article has mainly concentrated. Rather, the issue has a more legislative tone, as though the patentability of computer programs and algorithms is an open issue about which it would be appropriate to ask if patents are "good" or "bad" for the industry and for society. Predictions that patents may be harmful to the software industry, computer science, mathematics, or society as a whole have been quite frequent, even from some of the most well-known people in the software and computer science fields.[428]

---

rangement, and allocation of known effects within a program are not patentable subject matter. *An International Overview*, *supra*, at 9; *see also* H. HANNEMAN, *supra* note 37, at 177-84; recent European Patent Offices decisions cited *supra* note 351.

In short, other nations seem to have reached roughly the same conclusion as the Supreme Court in *Diehr* on the patentability of industrial process improvements involving programs; they have not, however, embraced the notion that all program innovations are patentable or that it is desirable to give up rules against patenting business methods, printed matter, mental processes, or mathematical methods in order to give software developers the kind of protection they might wish to have. Thus, if the United States is to be in harmony with other nations on its policy regarding computer program-related inventions, the shift back to the conservative view of *Diehr* is warranted.

[427] *See* sources cited *supra* note 16.

[428] *See, e.g.*, MIT Notes, *supra* note 16, at 14-16 (remarks of Dan Bricklin, co-developer of the first electronic spreadsheet program VisiCalc, critical of patents for software); *Kapor Testimony*, *supra* note 413, at 3-4 (questioning software patents); Plauger, *Soup or Art? Copyright Protection for Software Concepts*, COMPUTER LANGUAGE, Sept. 1989, at 17 (opposing patent protection for software innovations because of their mathematical character). Professor Newell's article, *supra* note 15, is also critical of patent protection for software.

See also Samuelson & Glushko, *supra* note 16, at 140 (survey of user interface field showing opposition to software patents); *Oversight Hearing on Computers and Intellectual Property Before the Subcomm. on Courts, Intellectual Property and the Administration of Justice of the House Judiciary Comm.*, 101st Cong., 2d Sess. (Mar. 7, 1990) (statement of the Software Publishers Ass'n) (noting that some software publishers regard patent protection as likely to discourage software development); LEAGUE FOR PROGRAMMING FREEDOM, AGAINST SOFTWARE PATENTS (forthcoming 1990) [hereinafter LEAGUE DRAFT]. The League's President and Founder, Richard Stallman, is a well-known computer programmer who recently received a MacArthur Foundation grant which he plans to use, in part, to support the League's fight against software patents.

Patent lawyers tend to respond to such concerns by saying that, although this may be an interesting social policy issue, the reality is that the legal system has already established a rule in favor of software patents. Court decisions have upheld the patentability of program-related inventions. There are many issued patents now, there will be many more in future years, and all are presumed to be valid. It would take congressional action to alter what has become the status quo, and that is not likely to happen.

This article argues that the legal foundation on which the current belief in the patentability of all program-related inventions rests is not as solid as many patent lawyers would like to believe. The CCPA decisions upholding software patents are hardly models of enlightened clarity and well-explicated principles. Indeed, the only principle which seems to have guided that court's deliberations over the years is one of upholding the patentability of as many program-related inventions as possible while still appearing to show some respect for the Supreme Court's decisions. Both the CCPA decisions and the Patent Office's current more generous attitude toward software patents go beyond what the Supreme Court approved in the *Diehr* case. As a result, the legal debate over the patentability of computer program-related inventions may not be as settled as some patent lawyers claim.

Although the author's study of the computer program patent cases has caused her to be critical of these decisions and of Professor Chisum's argument for overturning *Benson*, it is primarily because of the widespread concerns about the ill effects of patents from within the industry and the technical community that she has pursued this study questioning patent protection for computer program-related inventions.

The remainder of this article discusses four policy alternatives by which the law could deal with the software intellectual property protection issues. One possibility would be to rely on copyright law alone (or in conjunction with trade secret law) for the legal protection of computer programs, a solution which the author believes many in the software industry would prefer.[429] A second would be to confine patents for program-related

[429] *See, e.g.*, MIT Notes, *supra* note 16, at 14-16 (remarks of Dan Bricklin, approving of copyright and critical of patents for software); sources cited *supra* note 428. This view is consistent with the views of those in the user interface field who approve of copyright for source and object code, but

inventions to those which are part of traditionally patentable industrial processes and machines and to rely predominantly on copyright and trade secret protection for protection of other programs.[430] A third alternative would be to accept a vast expansion of the domain of patents and to try to define new roles for both copyright and patent law to play in the protection of programs (and perhaps a wide spectrum of other intellectual work).[431] A fourth would be to develop a sui generis system of legal protection for computer programs which recognizes that programs have characteristics that make it difficult to fit them successfully into the existing patent or copyright systems.[432] It is possible to design a law that is appropriate to the kind of subject matter that software is. This law would provide a suitable degree of protection for software innovations while defining an allowable scope for development of competitive products.

A. Reliance on Copyright Alone to Protect Program Innovations

The strongest point in favor of relying on copyright rather than patent law for the protection of computer programs is that the software industry has grown tremendously under the regime of copyright and has exhibited

---

do not support patent protection for program innovations. *See* Samuelson & Glushko, *supra* note 16, at 140. This alternative would be especially attractive to the software industry if decompiling or disassembling object code to discern the algorithms (or other trade secret material) it contains were made illegal. Although some have argued that these acts are already illegal under copyright law, others (including the author) argue that under copyright law, such acts are presently lawful. *See* sources cited *infra* note 483. Some part of the perceived need for patent protection for software innovations may arise from the vulnerability of software to such technical analysis. *See infra* notes 460-61 and accompanying text.

[430] This seems to be the option most consistent with the Supreme Court opinion in Diamond v. Diehr, 450 U.S. 175 (1981), *see supra* notes 260-67 and accompanying text, and with developments in the international arena, *see supra* note 426.

[431] This is the alternative most favored by patent lawyers. *See, e.g.*, Sumner & Lundberg, *supra* note 11. Chisum's article argues not only for the patentability of algorithms, but also for a wider scope of patent protection for other inventions. He advocates the abolition of the mental process, printed matter, and business methods limitations on patentability. *See supra* note 423.

[432] *See, e.g., Oversight Hearing on Computers and Intellectual Property Before the Subcomm. on Courts, Intellectual Property and the Administration of Justice of the House Judiciary Comm.*, 101st Cong., 1st Sess. (Nov. 8, 1989) (statement of Anne Wells Branscomb, Esq.); Samuelson, *supra* note 418; Menell, *Tailoring*, *supra* note 419; Karjala, *Lessons From the Computer Software Protection Debate in Japan*, 1984 ARIZ. ST. L.J. 53 (1984); Comment, *Softright: A Legislative Solution to the Problem of Users' and Producers' Rights in Computer Software*, 44 LA. L. REV. 1413 (1984); *see also* Reichman, *supra* note 334.

an astonishing capacity for creative innovation in the past two decades.[433] The fact that this growth has occurred without the aid of patent protection is powerful evidence that patent protection is not necessary for the software industry to thrive. In view of this, it is not unreasonable for software developers to perceive that injecting patents into the software protection landscape might disrupt the balance that has been achieved under the copyright (and trade secret) regime to which the industry has become accustomed.[434]

Many in the software industry believe copyright has a number of significant advantages over patents as a form of legal protection for programs. With copyright, all that a software developer must worry about is developing its software independently and not copying from someone else's product. With patents, the first inventor (or at least the first inventor to get to the Patent Office) can exclude competitors entirely from the market; thus, it may be decades before competitive application programs can enter the market.

The potential vulnerability of software products to late-issuing patents is of special concern to software developers and publishers. Software products often take several years to develop from a raw idea to a marketable final version. No matter how thoroughly or how often one searches the records of issued patents, one can never know when a patent affecting a software product might issue.[435] Developers worry that on the day they introduce their newest and most innovative product to the market, a patent might have issued to another firm that would cover some aspect of the just-introduced product, thereby jeopardizing the firm's investment in the new product. Thus, opponents of software patents liken them to "time-bombs" in the software development process.[436] Copyright poses less of a threat in this regard.[437]

---

[433] *See, e.g.*, OTA BACKGROUND PAPER, *supra* note 3, at 1.

[434] *See, e.g.*, *Will Software Patents Cramp Creativity?*, Wall St. J., March 14, 1989, at B1, col. 3; *see also* Kahin, *The Software Patent Crisis*, TECH. REV., April 1990, at 53. Even if neither patent nor copyright protection were available for algorithms, they could still be (and are) protected as trade secrets. The more obvious of them would be independently developed by other programmers; the less obvious could give their developers a competitive edge without the aid of the patent system.

[435] In addition, the Patent Office's classification system for program-related inventions may make it difficult to find already issued patents that may affect a software developer's product. *See, e.g.*, Kahin, *supra* note 434, at 55.

[436] *See, e.g.*, OTA BACKGROUND PAPER, *supra* note 3, at 9.

[437] Theoretically, it poses no threat at all, for independent development is a complete defense to a

Some software developers also worry that in a climate in which patents can issue on so many subcomponents of computer programs, a future software developer would have to obtain so many licenses from so many different firms for so many different components to make a particular computer program that the cost of developing software would be driven to prohibitive levels.[438] With such increased development costs, the price of software would inevitably rise for consumers, eventually causing the size of the market for software to be smaller than it might otherwise have been.

Because of the stronger monopoly right that they convey, patents do seem likely to increase barriers to entry significantly in the software market. One of the reasons that the CONTU Commission (on whose recommendation Congress relied in adopting a copyright regime for computer programs) regarded copyright as a superior legal regime to patents for this industry was that the Commission regarded copyright as likely to keep competitive barriers to entry low.[439] Because so much of the most innovative software products now available in the market have come from individuals who started off "in a cottage,"[440] it should be of serious concern

---

charge of copyright infringement. However, because it is possible for a jury to infer illicit copying from evidence of access to the copyrighted work and substantial similarity of content, *see, e.g.*, Arnstein v. Porter, 154 F.2d 464 (2d Cir. 1946), there is some chance that one who has, in fact, independently developed a similar work might nonetheless be found liable for copyright infringement.

[438] *See* MIT Notes, *supra* note 16, at 15; *see also infra* note 440.

[439] *See* CONTU FINAL REPORT, *supra* note 3, at 23-25; *see also* MIT Notes, *supra* note 16, at 7 (software patents would raise the entry barriers for small companies). However, some copyright infringement decisions, in construing the scope of copyright for software so broadly, suggest that the predictions that copyright will keep entry barriers low and competition keen seem overly optimistic. *See, e.g.*, Whelan Assocs., Inc. v. Jaslow Dental Laboratory, 797 F.2d 1222 (3d Cir. 1986), *cert. denied*, 479 U.S. 1031 (1987); Lotus Dev. Corp. v. Paperback Software, Inc., 740 F. Supp. 37 (D. Mass. 1990).

[440] Dan Bricklin, co-developer of the VisiCalc program, characterizes the software industry as "inherently cottage-based." MIT Notes, *supra* note 16, at 15; *see also* Kahin, *supra* note 434, at 55-56 (predicting a restructuring of the software industry if current patent trends continue). One reason that patent protection for software innovations has the potential to raise entry barriers is the cost of either building an in-house patent lawyer staff or hiring outside patent counsel to prepare and prosecute patent applications for the firm. Costs per patent application can exceed $10,000; even patent searches to make sure one is not going to tread on someone else's patent, though less costly than prosecuting patent applications, can be expensive and time-consuming and may require professional patent counsel. Defending any patent one gets can also be far more expensive than the cost of obtaining a patent. For start-up companies wanting to devote their resources to improvement and marketing of a product, the costs of hiring patent counsel and paying patent fees may make entry prohibitive, especially since they must largely be incurred well before one can realistically know whether the

that software patents may create entry barriers that would not have been created by copyright.

In addition, opponents of patents for software believe that the Patent Office has been issuing many very "bad" patents (ones that should not have issued because the inventions claimed were already in the state of the prior art or were trivial advances).[441] They believe that the Patent Office examiners have insufficient expertise about computer program technology in general, and the state of the art in this field in particular, to make good decisions on the novelty and nonobviousness of program-related claims.[442] Because of the contorted way in which computer program-related innovations have been handled in the patent system, the patent classification system is also regarded as inadequate.[443] These deficiencies make it difficult for the patent system to perform one of its most important functions as a repository of knowledge about the significant advances in technological fields,[444] enabling someone to make a meaningful search through the patent files to find out if a particular idea has been invented before. Because

---

product embodying the innovation will be a commercial success. *See, e.g., Letter to the Editor: Patents Aren't Panaceas*, BYTE, Nov. 1989, at 36, 38 (discussing problems small software firms see with software patents). It is no wonder, then, that the firms which have been the most active in patenting software-related innovations have been computer manufacturers, such as IBM Corp., that already have full-time patent lawyers on staff and a company culture favoring patents. *See* Soma & Smith, *Software Trends: Who's Getting How Many of What? 1978 to 1987*, 71 J. PAT. OFF. SOC'Y. 415, 421 (1989). One strong appeal of copyright (and trade secret) as a form of legal protection, from the software industry's standpoint, is the automatic protection it affords (without lawyers). Because of the societal (as well as individual) costs of the patent system, we should, at a minimum, do a cost-benefit assessment before embracing the use of the patent system for software-related inventions. It is far from clear that the benefits of applying the patent system to software would outweigh the costs.

[441] *See, e.g., Kapor Testimony*, *supra* note 413, at 3-4.

[442] *See, e.g.*, LEAGUE DRAFT, *supra* note 428; Kahin, *supra* note 434, at 55; OTA BACKGROUND PAPER, *supra* note 3, at 8-9. Even the author of a recent law review note arguing in favor of patents concedes the need for high standards for nonobviousness determinations. Note, *supra* note 294, at 1063. It is worth noting that the Patent Office does not presently accept degrees in computer science (except one concerning computer circuitry), mathematics, or software engineering as an appropriate technical background for admittance to the patent bar. *See* U.S. PATENT & TRADEMARK OFFICE, GENERAL REQUIREMENTS FOR ADMISSION TO THE EXAMINATION FOR REGISTRATION TO PRACTICE IN PATENT CASES BEFORE THE U.S. PATENT AND TRADEMARK OFFICE 2 (1990) (computer engineering is the only computer-related technical degree accepted). If software patents are to continue to be issued, it should at least be possible for those with real expertise in these fields to be among those eligible to prosecute patent applications for computer program-related inventions.

[443] *See, e.g.*, OTA BACKGROUND PAPER, *supra* note 3, at 8-9.

[444] *See, e.g.*, Bonito Boats, Inc. v. Thunder Craft Boats, 489 U.S. 141 (1989) (striking down Florida "plug mold" statute in part because of potential for interference with patent system as a repository for technological innovation).

the Patent Office, for the most part, did not accept program-related claims for the first thirty years or so of the field's existence, it will be difficult for the Patent Office to be brought "up to speed" on computer program-related innovations in order to serve this repository function.[445]

In addition, many mathematicians, computer scientists, and others in the software development community assert that computer program algorithms ought not to be protected by intellectual property law because they are, as the Supreme Court first said eighteen years ago, basic intellectual tools of science and mathematics.[446] Algorithmic advances have been extremely important to growth in the field of computer science, one of the most fecund fields during the latter part of the twentieth century, and it has grown without patent protection. As Professor Newell observes, if we want to be sure that fundamental theoretical discoveries of computer science are available to be used, tested, and built upon by others, it may be more sensible to withhold patent protection for algorithms than to grant them.[447]

Certainly mathematicians and computer scientists would scoff at the CCPA's effort to distinguish between "mathematical algorithms" and non-mathematical algorithms.[448] Algorithms for computer programs are mathematical by their very nature. Programs are instantiations of algorithms, which direct the performance of logical operations on data by the computer. The CCPA has had a mistaken understanding of the nature of

---

[445] *See, e.g.*, Galler, A Proposal for a Software Patent Institute (June 26, 1990) (proposing to create a prior art data base to aid Patent Office determinations).

[446] *See, e.g., Mathematicians Are Troubled by Claims on Their Recipes*, N.Y. Times, March 12, 1989, at E26, col. 1. Although Professor Chisum disparages the Supreme Court's *Benson* decision for arguing that algorithms such as Benson's should be unpatentable as "basic tools of scientific and technological work," Chisum, *supra* note 12, at 980-83, he also recognizes that "academic" uses of such algorithms may be important to the growth of the field of computer science and proposes a "fair use" defense, along copyright lines, to cover such situations. *Id.* at 1018-19. Others might argue that the "experimental use" limitation on the scope of patent rights should shield academic uses from patent infringement liability. *See, e.g.*, Eisenberg, *supra* note 331.

[447] *See supra* notes 333-34 and accompanying text.

[448] *See supra* notes 389-94 and accompanying text. Among the issues that the new Office of Technology Assessment study of computer software protection will address is the need for a clear and uniform definition of terms like "algorithm" so that legal decisionmakers will be able to use the term in appropriate ways and not make it the basis of unworkable distinctions. *See* OFFICE OF TECHNOLOGY ASSESSMENT, OTA PROJECT PROPOSAL, STAYING ON TOP: THE CHALLENGES OF TECHNOLOGICAL CHANGE & GLOBAL COMPETITION IN PROTECTING INTELLECTUAL PROPERTY TABLE 1 (April 18, 1990).

mathematics (tending to mistake it for numerical calculations), and that of computer programs (generally not recognizing the data representation, organization, manipulation, and display processes they embody in copyrighted texts). If the patent system is unable to conceptualize computer programs appropriately, it should leave program protection to other protection systems.

Adopting copyright as the federal law governing intellectual property protection for software would also eliminate the need to sort out what may turn out to be an unworkably complex problem of defining the patent/copyright interface for computer programs.[449]

B. *Limiting Software Patents to Traditional Industrial Processes and Machines*

The second alternative consists of relying on copyright as the predominant form of intellectual property protection for computer programs and defining a role for patents in protection of certain program innovations, such as those that are elements in traditionally patentable industrial processes and machines. It is reasonably consistent with the first alternative's reliance on copyright and trade secret as the forms of intellectual property protection for computer programs. This alternative, in effect, would be a renewal of the Patent Office's first policy regarding program-related claims and an implementation of the Supreme Court's intent in *Diamond v. Diehr*.[450] It is, moreover, consistent with the long tradition of the patent system of interpreting the meaning of the term "process" as those involving transformation of matter. One of the advantages of this approach would be in maintaining the integrity of traditional patent subject matter boundaries.[451]

An additional advantage of this approach is that it seems the most consistent with the direction in which the European community, among others, seems to have moved in the protection of computer software innovations.[452] There is much to be said for a stable international scheme for

---

[449] *See infra* notes 467-74 and accompanying text.

[450] 450 U.S. 175 (1981); *see supra* notes 260-81 and accompanying text.

[451] *See supra* notes 409-26 and accompanying text.

[452] *See supra* note 426.

the protection of software innovations.[453] The United States will benefit more in the long run from harmonizing its laws and its practices under those laws with those of other nations than from proceeding in its own direction, particularly as to a subject matter like computer software, which has such promising prospects as a major export item for U.S. industry.

However, there would still be problems with implementing this alternative. First, the Patent Office would have to be persuaded to adopt a narrower interpretation of the subject matter provisions of the patent statute and to reeducate some of its examiners so that information processing patents, like some of those granted in recent years, do not issue.[454]

Another problem with both this and the first alternative is the question of what to do about the patents already issued for what would now be "unqualified" subject matter. The Patent Office would need to review these already-issued software patents and make a redetermination of patentability. Then, if any of them are not patentable, the Patent Office would declare them invalid and unenforceable. Even assuming that one could overcome these obstacles, there would still be administrative difficulties in implementing this rule. Patent lawyers have become adept at claiming software innovations in such a way that one sometimes wonders whether examiners fully understand the claims for which they are issuing patents.[455]

Even if patent lawyers could be persuaded to claim software-related innovations in a straightforward manner, there would continue to be many situations in which line-drawing by patent examiners would be dif-

---

[453] If intellectual property protection became part of the General Agreement on Tariffs and Trade (GATT) system, there would be significant pressures on member nations to harmonize their domestic protection systems to conform to international norms when protection disputes arose. *See generally Trade-Related Aspects of Intellectual Property*, 22 VAND. J. TRANSNAT'L L. 223-384, 689-922 (1989) (appearing in issues 2 and 4); Reichman, *Intellectual Property in International Trade: Opportunities and Risks of a GATT Connection*, 22 VAND. J. TRANSNAT'L L. 747 (1989). Even if GATT does not become a system for resolving conflicts over intellectual property protection, there will still be increasing need for international agreement and coordination of efforts to protect intellectual property in a harmonized manner. *See, e.g.*, OTA REPORT, *supra* note 9, at 213-53.

[454] *In re* Meyer, 688 F.2d 789 (C.C.P.A. 1982), and *In re* Grams, 888 F.2d 835 (Fed. Cir. 1989), are signs that the Patent Office has neither opened the patentability gates as wide as many patent lawyers have been asserting nor as wide as some patents that have issued in recent years might have suggested. *See supra* notes 367-76 and accompanying text.

[455] It seems unlikely, for example, that the examiner of the Pardo patent understood the claims to include the natural order recalculation function for spreadsheet programs. *See supra* note 249.

ficult. However, this is not unusual in the law. Any time one creates subject matter boundaries, close cases will have to be decided. The Patent Office was once confident of its ability to draw such lines, and patent offices abroad are doing so. Hence, if there is a will to bring this alternative into practice in the United States, it can be done.

C. *Accepting a Vast Expansion of Patentable Subject Matter and Working to Define an Appropriate Patent/Copyright Interface for Computer Programs*

Patent lawyers, of course, would prefer the third alternative, which they perceive to be the current trend of patenting all manner of software innovations. Their professional training leads them to believe that wherever the patent regime spreads its net, progress and innovation are bound to follow. In addition to pleasing the patent bar, this option would obviously please those persons and firms that now look to patents to protect their software innovations.

One argument for this third alternative is that it seems to many to be the status quo.[456] The forces that favor a pro-patent status quo must be powerful because they seem to have overcome what had previously been the status quo of "no patents." The author, however, does not believe that one should accede to the status quo merely because it is the status quo and questions whether it is as much the status quo as some assert that it is. For the purposes of a balanced presentation of the public policy issues, she will, nevertheless, articulate a number of arguments favoring patent protection for algorithms and a wide range of other inventive aspects of computer programs, even though it would require "stretching" the patent system to do so.

The most powerful argument against patents for program-related inventions is that the industry has grown from being a nonexistent industry to a major, flourishing, and highly innovative industry without patent protection. It is no answer to say, as some patent lawyers might be tempted, "Just think how innovative it would have been had patents been in place." The absence of patents allowed the rapid spread of ideas about

---

[456] *See, e.g.*, Sumner & Lundberg, *supra* note 11.

programming computers. The experimentation with, implementation of, and modification of these ideas have been critical factors in both the growth of the computer science field and the growth of the computer software industry, which has utilized the ideas spawned from this science.[457] In the mid-1960s, leading scientists and industry representatives judged that patents were not desirable for program innovations, and leading computer firms (who were at that time also among the leading developers of computer programs) filed amicus briefs that opposed patents for program algorithms.[458] Thus, let us accept as a working assumption that the computer software industry has become a major industry without the aid of patents, and that had patents been in place in the industry's infancy, the field would not have grown as it has.

It may be, however, that the conditions which promoted innovation in this field in the early phase of its development are not the same as those that will promote it in a later phase. If the absence of patents was "good" for the industry in its infancy, it does not necessarily follow that at a different stage of its development, patents would be "bad." Changed conditions, then, might well be a strong argument in favor of a broadened scope for patents in the software field.[459]

Second, it may be that there are certain kinds of innovations in software that once were protected as trade secrets (algorithms, for example), but which have now become less suitable for trade secret protection as software products have become more like other commodities in the market and as more sophisticated "reverse engineering" or "re-engineering" tech-

---

[457] *See supra* note 334 and accompanying text.

[458] *See supra* note 90. Although the nature of program algorithms has not changed since 1972, the position of those who filed amicus briefs against a patent for the Benson algorithm does seem to have changed. IBM, Honeywell, and Unisys (successor company to Burroughs) now seek patents for their software innovations. *See* Soma & Smith, *supra* note 440; *see also Oversight Hearing on Computers and Intellectual Property Before the Subcomm. on Courts, Intellectual Property and the Administration of Justice of the House Judiciary Comm.*, 101st Cong., 1st Sess. 7-8 (Nov. 8, 1989) (statement of Computer and Business Equipment Manufacturers Ass'n) (expressing satisfaction with current software patent practices). CBEMA too was once an amicus curiae against the Benson patent application. *See supra* note 90.

[459] No one has yet made a case for such changed conditions, but perhaps it could be made. In the absence of some evidence of a change warranting patent protection, it would be more prudent to stick with what has brought real growth and prosperity to the field and the industry — namely copyright — unless, of course, copyright law becomes so expansively interpreted that, over time, it comes to retard progress in the software field more than patent protection would. *See supra* note 439 and *infra* note 462.

nology has become available.[460] Then, patents may fill an important legal protection gap as trade secrecy recedes in importance in the software field.[461]

Third, it may be that copyright law, properly construed by traditional doctrines which limit its scope, is really not suited to provide the extent of protection for computer programs necessary to support the industry's continued growth in the long run. There may well be some features of computer programs (algorithms, among them) which traditional copyright law would not protect, but which, under present conditions, need to be given some protection to create the right kind of incentive for investment in the software industry. If the choice is made against developing a new law exclusively for computer programs, this question arises: Should one protect these features through copyright or patent law?[462]

It may be unwise, as an economic matter, to give the lengthy protection provided by copyright to such innovations as program algorithms that meet only copyright's minimal "originality" standard.[463] The patent paradigm may be more appropriate because it provides stronger protection, but the protection is of shorter duration and is limited to innovations which

---

[460] *See, e.g.*, Chikofsky & Cross, *Reverse Engineering and Design Recovery: A Taxonomy*, IEEE SOFTWARE, Jan. 1990, at 13.

[461] One further argument patent lawyers could make in favor of patent protection for program innovations is that if patents were available for such things as algorithms, software firms would likely be more willing to disclose their algorithms instead of holding them as trade secrets. *See In re* Sarkar, 588 F.2d 1330, 1332-33 (C.C.P.A. 1978).

Professor Reichman argues that there is a close kinship between the protection problems now faced by the software industry and those presented by other hybrid innovations. *See, e.g.*, Reichman, *supra* note 415. Full-fledged patent law, he argues, is not the only way to "fill the gap" in legal protection for these kinds of innovations.

[462] Professor Chisum points out that unless patent protection is available for certain kinds of program innovations, such as algorithms, there will be a strong temptation to "fill the gap" by construing copyright to cover algorithms. *See* Chisum, *supra* note 12, at 1020. Indeed, some have argued that program algorithms can and should be given protection by copyright law. The *Whelan* case can be read to support this interpretation, for it says that *everything* about a program but its general purpose or function can be protectible expression under copyright law (unless there is only one way to perform the function). *See* Whelan Assocs., Inc. v. Jaslow Dental Laboratory, 797 F.2d 1222, 1235-40 (3d Cir. 1986), *cert. denied*, 479 U.S. 1031 (1987). Since algorithms can provide the organizational framework for a computer program, one could argue that algorithms are part of the "structure, sequence, and organization" of the program which copyright can protect. *See, e.g.*, Abrams, *Statutory Protection of the Algorithm in a Computer Program*, 9 COMPUTER/LAW J. 125, 143 (1989). *But see* Note, *Idea, Process, or Protected Expression*, 88 MICH. L. REV. 866 (1990).

[463] *See, e.g*, Menell, *An Analysis*, *supra* note 419, at 1080.

are not already in the state of the art (those meeting patent's novelty requirement) and which represent significant advances in the art (those meeting patent's nonobviousness requirement).[464] Patent protection may fill the protection gap left when traditional copyright principles are applied to program abstractions.[465]

A fourth reason to accept a broadened scope of patent protection for program-related inventions might be that computer program innovations are, by their very nature, different from the types of technological innovations traditionally thought of as "industrial."[466] If the nature of technological innovation has changed, then so perhaps should the law of patents. There is certainly as strong a set of reasons to give incentives to those who contribute to the growth of this new field of technology as there is to provide incentives to those whom the traditional patent law serves. Although it undoubtedly requires a "stretch" to protect computer program-related innovations by patent law, the subject matter stretch that would be required for patent protection is less than the stretch that has already occurred in protecting those innovations through copyright. As long as relatively clear and workable boundaries between patent and copyright law are defined and maintained, and as long as judges in copyright cases acknowledge and make accommodations for the technological nature of computer programs, it may be that patent and copyright law can evolve to provide meaningful and appropriate intellectual property protection for computer programs, thereby balancing the need to protect innovation with the need to promote competition in the development of innovative products.[467]

---

[464] *Id.*

[465] It is ironic that in order to preserve something which approaches the traditional bounds of copyright law, one would need to overstretch traditional bounds of patent law, but it seems that we cannot protect computer software to the degree necessary except by somewhat "distorting" both copyright and patent doctrines.

[466] Professor Newell observes, "It is entirely conceivable that some areas of technology will end up with *all* of the inventive activity in terms of algorithms. That is, with the transfer of creativity . . . even though a domain involves transformations of matter, yet all inventions of useful transformations occur by the invention of algorithms." Newell, *supra* note 15, at 1030.

[467] *See, e.g.*, Maier, *Software Protection — Integrating Patent, Copyright, and Trade Secret Law*, 28 IDEA 113 (1987) (optimistically predicting that each law will have an appropriate niche in the protection of program innovations); *see also* Samuelson, *Look and Feel*, *supra* note 14; Samuelson, *Right Course*, *supra* note 14, for the author's effort to aid in achieving an accommodation of computer programs in a copyright and patent framework. But see *infra* notes 470-74 and accompanying text for some questions about defining the patent/copyright interface for programs.

In order to respond to Patent Office concerns about its ability to handle computer program-related innovations,[468] a scientific advisory board could be established to advise the Office on how to overcome past problems.[469] This advisory board could help the Patent Office develop patentability distinctions that would be meaningful in the software development community so that the community would gain confidence in the Office's decisionmaking.

This board could also help the Patent Office identify aspects of computer programs that need patent or patent-like protection. Then, perhaps the patent bounds could be expanded to provide protection to these kinds of features. However, this expansion should not extend as far as some patent lawyers favor. They claim that every aspect of software should be patentable because drawing narrower boundaries is too difficult.

Certainly before Professor Chisum's extremely broad view of patentable subject matter is adopted, there should be a full airing of the public policy implications of such an adoption, with, at a minimum, the supervision of Congress. Because intellectual property lawyers do not presently agree about the application of patent and copyright law to computer programs[470] or about the relationship between these two laws in the protection of software,[471] these issues should also be reviewed by the legislature.[472] Without a consensus on these matters,[473] intellectual property

---

[468] *See supra* notes 441-45 and accompanying text.

[469] *See, e.g., Oversight Hearing on Computers and Intellectual Property Before the Subcomm. on Courts, Intellectual Property and the Administration of Justice of the House Judiciary Comm.*, 101st Cong., 2d Sess. 23 (Mar. 7, 1990) (statement of Daniel Bricklin); *see also* Galler, *supra* note 445.

[470] A symposium issue of the AIPLA Quarterly Journal was devoted to this topic. *See Computer Programs: The Patent/Copyright Interface*, 17 AIPLA Q.J. 173 (1989). In this issue, the author reported the results of her survey of the intellectual property lawyers on an AIPLA subcommittee. The subcommittee members were asked for their opinions on which aspects of programs copyright protects, which aspects patent protects, what degree (if any) of overlap the lawyers perceived in patentable and copyrightable aspects of software, and what implications they thought flowed from the patent and copyright boundaries. *Samuelson Survey*, *supra* note 14. Lawyers responding to the survey disagreed among themselves about the scope of copyright and patent protection for program innovations, particularly as to internal structural abstractions of programs. Overruling *Benson* and further widening the bounds of patent law would likely exacerbate these problems.

[471] *See, e.g., Samuelson Survey*, *supra* note 14 (showing a deep split among the survey respondents about the extent to which the subject matters of copyright and patent overlap in the protection of program innovations).

[472] The author is among those who regard the domains of copyright and patent law, insofar as both laws extend to computer software, to be separate and exclusive. If, for example, patent protection

law, as applied to computer software (and perhaps even as applied to other innovations) may be in danger of becoming more chaotic and uncertain.[474] This result would not be in the best interest of the software industry.

---

were available for the algorithm underlying a program, then copyright would not be available. If a patent were to issue for a particular algorithm, the patentee's permission would be needed to implement the algorithm in a program. After the patent expired, others could use the algorithm, but would have to write their own code to implement the program rather than copying the ex-patentee's copyrighted code.

Even under a regime of this type, a great many questions would still be raised by the application of both copyright and patent to programs. These questions include: Would drawing a flow chart for a program to implement a patented algorithm infringe the patent? Would writing source code for a program implementing the algorithm infringe? Or would a patent on an algorithm be infringed only when the source code is converted to object code? If the Patent Office requires source code to be disclosed to satisfy the best mode requirement, would that code become public domain when the patent expires? Or could that code only be copied from the patent itself? Or will others have to write their own code even though the inventor has revealed this code as the best mode of implementation?

[473] It is sufficient to give one illustration of the problems likely to arise in the future if both patents and copyrights are applied to the same aspects of programs. Patents have been issuing for data structures for computer programs. *See, e.g., In re* Bradley, 600 F.2d 807 (C.C.P.A. 1979), *aff'd by an equally divided court sub nom.* Diamond v. Bradley, 450 U.S. 381 (1981). Would a program data structure that was patented also be protected by a copyright in the program so that when the patent expired, the copyright would "kick in" to provide several extra decades of protection against copying? In general, when a utility patent expires, the innovation falls into the public domain and is available for free copying. Sears, Roebuck & Co. v. Stiffel Co, 376 U.S. 225 (1964).

Suppose the data structure were not really novel or obvious. Could it then be protected by a copyright, by virtue of meeting copyright's low originality standard, for several decades longer than if it had met the patent nonobviousness and novelty standards? Patent law has traditionally considered that an innovation that was not novel or nonobvious enough to be eligible for patent cannot be protected (unless it can be kept as a trade secret), and thus, is available for free copying. *Id.* If copyright came to cover unpatented but patentable features of a computer program, the public would lose the free availability for copying that patent law has traditionally assumed for unpatented items. *See Samuelson Testimony, supra* note 14.

[474] The author has argued elsewhere that one of the difficulties in resolving the patent/copyright interface for computer programs is the "turf war" between patent and copyright lawyers over computer program protection issues. In its approach to program protection, each group has a strong economic interest in a broad construction being given to the bounds of its specialty and a correspondingly narrow construction being given to the other law. *See, e.g.*, Samuelson, *supra* note 418.

Also contributing to the difficulty of sorting out the boundaries of patent and copyright are the differing conceptual constructs with which patent and copyright lawyers tend to analyze computer program protection issues. When a patent lawyer looks at a computer program (perhaps even down to the level of code), he sees a "machine" (or machine design) or "process" for operating a machine; when a copyright lawyer looks at a computer program, she tends to see a "writing" (that is, a work of applied art or literature) and protectible expression, including structural features of the writing. This brings to mind the adage that when the only tool you have is a hammer, everything begins to look like a nail.

D. Sui Generis Legislation for the Protection of Computer Programs

Accepting an expansive realm for software patents requires ignoring that many in the software industry itself are strongly opposed to patents for software innovations. This warning of harm should be of concern to us all. The point that innovation in the software field has developed rapidly without the aid of patents should not be forgotten. If those in the industry who now seek patents could show that conditions have changed so much that what would once have been harmful would now be beneficial, then there might be more reason to expand the existing patent realm without this showing.

Absent this showing, either copyright alone or a fourth alternative should be adopted: the construction of a new law for the protection of computer programs and the innovations they embody. If it is true that computer programs are too much of a machine to fit comfortably in the copyright system and too much of a writing to fit comfortably in the patent system, it would be better to create a new law for the protection of computer programs than to stretch existing legal categories past their breaking points.

Many agree that patent law may be unsuited for the protection of program innovations. For example, Professor Newell argues that the conceptual model on which the patent system is built is "broken" and that a new model must be constructed — one that is attentive to the nature of programs and of the "inventables" which computers permit people to create and utilize.[475] In a recent article, Professor Reichman suggests a conceptual framework for the protection of computer programs as an instance of applied scientific know-how.[476]

The first step in creating a sui generis statute for the protection of computer program innovations would be the development of an appropriate conceptual model about both the nature of the subject matter to be protected and the purposes to be served by a protection statute. While there are similarities between software and other legal hybrids which Professor Reichman identifies as "applied scientific know-how," computer programs have unique characteristics which suggest that a more specialized treat-

---

[475] *See* Newell, *supra* note 15, at 1033-34.

[476] *See* Reichman, *supra* note 334, at 652-67, 714-18.

ment of programs alone would be more appropriate.

The Semiconductor Chip Protection Act of 1984 [hereinafter "SCPA"] is an example of a law specially created for the protection of a technology that did not "fit" comfortably in the patent or copyright systems. Even though computer programs and semiconductors are closely related technologies, one cannot merely extend SCPA's coverage to programs and thereby solve all the accompanying legal protection problems. Programs present different, and arguably more complex, protection problems from semiconductors. This fact does not mean, however, that the SCPA cannot be a useful model for the creation of a sui generis law for programs, at least as to some of its features. As with the SCPA, it would be desirable for computer program protection law to borrow some features from copyright law and patent law and to develop rules for protection problems that arise from the nature of programs themselves.[477]

At two workshops on software intellectual property law that were held in the fall and early winter of 1989 and sponsored by the National Research Council, Dr. Robert Spinrad of Xerox's Palo Alto Research Center suggested some basic principles (which he called the "five C's") that he thought a law for the protection of computer programs should satisfy.[478] One principle was assured "coverage." There were two kinds of coverage he favored: coverage for the product of hard work and coverage for brilliant work. His second principle was "continuity." As important as protection for his firm's own products might be, it was also important that his firm's development efforts be able to build upon existing standards and conventions and not have to develop everything from scratch. Spinrad's third principle, "consistency," could also be called "certainty." He wanted a legal protection scheme that would predict, with relative certainty, both the amount and duration of protection for an innovation. His fourth principle, "cognizance," involved having timely awareness of the intellectual property rights of others as they might affect his firm's products. Spinrad's main concern with patent protection was the potential for get-

---

[477] *See* Samuelson, *Creating a New Kind of Intellectual Property: Applying the Lessons of the Chip Law to Computer Programs*, 70 MINN. L. REV. 471 (1985).

[478] National Research Council, Computer Science and Technology Board, Workshops on Software Intellectual Property Law, September 11-12, 1989, and December 1-2, 1989. A book on the proceedings at this workshop will be published by the National Academy of the Sciences Press later this year. *See* REPORT OF THE FORUM ON INTELLECTUAL PROPERTY ISSUES IN SOFTWARE (forthcoming 1991).

ting "blind-sided" by a late issuing patent. Finally, he wanted "convenience." The process of obtaining the rights should be simple, inexpensive, and unlikely to lead to litigation.

Taking these principles as a starting point, a sui generis law for computer programs might include the following features:

> 1) Certain coverage for minimally "original" program source and object code from the moment of fixation in either form (Spinrad's "hard work" coverage);
>
> 2) An exclusive right to transform source into object code, and an exclusive right to control certain unauthorized copying and distribution of source or object code;
>
> 3) Protection for object code such that if an unauthorized person mechanically transformed that code into an object code which was not identical to the protected code, but was proven to be merely a "scrambled" version of it, infringement would be found;
>
> 4) Protection for source code that would protect against direct translation of that code from one programming language to another programming language in a related class of languages;
>
> 5) A duration of protection limitation that would commence upon the first commercial or other public distribution of the code and would last no more than twenty years;
>
> 6) A mandatory notice of assertion of protection of the program code which would include the year of the first commercial or public distribution of the code;
>
> 7) A copyright-like registration process (low cost, fast, minimal review) for program code; and
>
> 8) A right on the part of lawful possessors of the protected code to modify and copy it in order to make it useful to themselves.[479]

To the extent the program caused the display of copyrighted material, that copyrighted material would still be protected, even after the right in the program for its display had expired.

Several other matters remain to be discussed: First, what, if anything,

---

[479] Since the issues addressed by a sui generis law are numerous, only a general outline of necessary features is offered here. *See also* Samuelson, *supra* note 477.

should be done about protecting "brilliant" work? Second, what should be done about the related problems of reverse engineering of program code, interface protection, and disclosure of source code?

One question concerning "brilliant" work is: What kind of "brilliant" work in the development of computer programs should be protected? While members of the software industry and others in the technical community should be the ones to identify the kinds of brilliant work needing protection, the most obvious candidate for protection is program algorithms. One possibility is to use a modified patent approach to protect such things, but to create other mechanisms for implementing a protection scheme for brilliant algorithms. If the model of invention of the existing patent system is "broken," and if thirty years of failing to keep up with the state of the art raises questions about whether the Patent Office could ever "catch up," then perhaps one of the prominent computer societies (such as IEEE or ACM) could serve as a conduit through which claims to new and unobvious algorithms could be channeled.[480] These societies have established classification systems for computer software innovations. They often do blind reviews of articles claiming to report on advances in the field of computing. Their reviewers know the state of the prior art, and they repeatedly make judgments about whether a claimed dramatic advance in computing is what it purports to be. Such a society could establish a blind review system for algorithms, and even if it did not issue certificates granting legal protection, it could at least verify the claims for a government office handling legal protection of program matters. Another alternative would be a "petty-patent" system whereby one could register one's claim of a novel advance, but be left to establish its novelty and/or nonobviousness in litigation.[481] Still a third modified patent approach would be to allow registration of claimed inventive advances, but to provide protection only against copying by others and not against those who independently develop the same thing.[482]

Other issues of particular concern to the software industry are "reverse

---

[480] *See* Galler, *supra* note 445 (recommending the establishment of a panel of 200 software experts to advise the Patent Office on patent applications for software inventions).

[481] *See, e.g.*, Note, *Petty Patents in the Federal Republic of Germany: A Solution to the Problem of Computer Software*, 8 SW. U.L. REV. 888 (1976).

[482] *See, e.g.*, Reichman, *supra* note 415 (recommending a modified copyright approach to protection of new technology innovations).

engineering" of program code (which some think should be lawful and others think should be unlawful)[483] and protection of interfaces (which some think should be protected and others think should be unprotected).[484] These issues are connected because much of the reverse engineering of software has been done to discover internal interface information.[485]

This knot of problems has a number of possible solutions. One solution is to require disclosure of interface information, and then either to make interfaces public domain material or to require low-royalty compulsory licensing for use of interface information while prohibiting reverse engineering of program code. Another solution is to allow reverse engineering, but only in order to discover interface information. Still another solution is to require disclosure of source code material (thus making reverse engineering unnecessary) while providing protection against copying to the applied scientific know-how that it embodies.

An advantage of the sui generis approach is that it could be devised to provide the degree of protection which the computer software industry claims is needed to create the proper incentives for investment. This objective could be achieved without distorting patent or copyright law or doctrine in an attempt to reconcile the conflicts between copyright and patent law involved in the protection of software.

Apart from the conceptual integrity that a sui generis law would finally introduce to intellectual property law as applied to computer programs, a key advantage of sui generis legislation is that it would be accompanied by an increased certainty in the field about what is and is not protected and under what conditions protection is available. It is unfortunate that so

---

[483] *Cf.* Grogan, *Decompilation and Disassembly: Undoing Software Protection*, COMPUTER LAW., Feb. 1984, at 1 (arguing that virtually all reverse engineering of software is and should be illegal); Samuelson, *Reverse Engineering Someone Else's Software: Is It Legal?*, IEEE SOFTWARE, Jan. 1990, at 90 (arguing that the illegality of reverse engineering depends on incorporating expression in the resulting program and not on the reverse engineering process).

[484] *Cf.* Lake, Harwood & Olson, *Opinion: Seeking Compatibility or Avoiding Development Costs? A Reply on Software Copyright in the EC*, 11 EUR. INTELL. PROP. REV. 431 (1989) (arguing that interfaces should be protected by copyright); *LaST Frontier Conference Report on Copyright Protection of Computer Software*, 30 JURIMETRICS J. 15, 21-22 (1989) (discussing problems arising from copyright protection for internal interfaces); *see also* Samuelson & Glushko, *supra* note 16 (user interface designers oppose both copyright and patent protection for user interface features).

[485] *See, e.g.*, Opinion, International Business Mach. Corp. v. Fujitsu Ltd., American Arbitration Ass'n, Commercial Arbitration Tribunal, Case No. 13T-117-0636-85 (Sept. 15, 1987).

many intellectual property lawyers are content to wait decades for the existing laws to be applied in a case-by-case fashion so that the contours of the law of software protection will be revealed by the "genius" of the common law.[486] Firms in the software market today are probably less than pleased at the prospect of being the guinea pigs in this common law process and want to know the answers to protection questions now. Unfortunately the answer they get now will depend on which lawyer they ask.

The sui generis alternative has two chief disadvantages. First, after persuading most other nations to adopt copyright as a form of protection for computer programs, it would be awkward for the United States to turn against copyright for programs on the international scene and to persuade others to adopt yet another system of protection. Second, intellectual property lawyers in the United States will likely oppose sui generis legislation for computer programs vigorously; the lawyers are largely content with the status quo and with the prospect of litigation as a way of deciding which aspects of software are protectible by what law. It would take a consensus of the software industry itself to create the political climate in which sui generis legislation could be successfully enacted. To reach this consensus, those in the industry would have to stop listening to their lawyers and start thinking about how the law should be shaped based on their experience.

VIII. CONCLUSION

The path of least resistance is to let the law evolve as it will and not to try to guide its development. Perhaps new subject matter boundaries for the patent system will be developed eventually. In time, the contours of patent and copyright as applied to computer programs will undoubtedly fill out. Eventually either the United States will have persuaded other nations to adopt its broad approach to patents for software, or it will modify its own approach. The author is no soothsayer and offers no predictions about what the future will hold. She hopes only that this article will help

[486] *See, e.g.*, Davidson, *Common Law, Uncommon Software*, 47 U. PITT. L. REV. 1037 (1986) (since common law has successfully evolved in other areas of the law, quests for new legislative systems of software protection should be avoided).

the evolutionary process along because the state of the law clearly needs a firmer footing than the one it currently has.