

ESCAPING THE WORLD OF “I KNOW IT WHEN I SEE IT”: A NEW TEST FOR SOFTWARE PATENTABILITY

*Brooke Schumm III**

Cite As: Brooke Schumm III, *Escaping the World of “I Know It
When I See It”: A New Test For Software Patentability*,
2 MICH. TELECOMM. TECH. L. REV. 1 (1996)
available at <<http://www.mtlr.org/voltwo/schumm.pdf>>

I. INTRODUCTION

A. *Synopsis of Issue*

“[N]o person, not even the most learned judge much less a layman, is capable of knowing in advance of an ultimate decision in his particular case by this Court whether certain material comes within the area”¹ The last words of this famous quotation are not “of software patents”; they are “of obscenity.” Justice Brennan adopted these words from Justice Black’s vigorous condemnation of the regulation of obscenity. In doing so, Justice Brennan turned away from the “without socially redeeming value” standard of obscenity that he had created in earlier opinions.²

While these words were written before software patents were a significant reality, the language seems like the most reasoned explanation of the swirl of conflicting case law surrounding the patentability of software, the recent issuance of proposed regulations of software patents by the Patent and Trademark Office (PTO), and the “sure” view that algorithms should not be patentable.

If a court does not desire an invention to be patentable, the court need only reach for the shibboleth of abstract formula, scientific truth, or natural law,³ and, presto, it is not patentable. If patentability is desired, the court need only seize the shibboleth of structure or physical

* Brooke Schumm III is licensed as a patent attorney and is a principal of the firm of Daneker, McIntire & Davis, P.C. in Baltimore, MD. He graduated with a degree in Operations Research/Industrial Engineering from Cornell University and with a J.D. from the University of Michigan Law School.

1. *Paris Adult Theatre I v. Slaton*, 413 U.S. 49, 87 (1973) (Brennan, J., dissenting).

2. *Id.* at 78–80 (criticizing *Roth v. United States*, 354 U.S. 476 (1957)).

3. *Mackay Co. v. Radio Corp.*, 306 U.S. 86, 94 (1939), *cited in* *Gottschalk v. Benson*, 409 U.S. 63, 67–68 (1972), *Diamond v. Diehr*, 450 U.S. 175, 188 (1981), and *Arrhythmia Research Technology Inc. v. Corazonix Corp.*, 958 F.2d 1053, 1057 (Fed. Cir. 1992).

aspect, and, however thin the connection, the inventor has a patent. As one author put it, "The . . . test seems to boil down to the unhelpful 'I know an abstract idea when I see it.'"⁴ Such convolutions bring to mind Justice Brennan's later comment in the *Paris Adult Theatre I*⁵ case that "one cannot say with certainty that material is obscene until at least five members of this Court, applying inevitably obscure standards, have pronounced it so."⁶ He was, of course, speaking of the Supreme Court and obscenity, not of the Federal Circuit and software patents.

Of particular concern in this article is the patent contract that the People of the United States ordained in the Constitution. The People provided for their benefit that in return for a disclosure of a useful invention, the People would grant the inventor a monopoly to exclude others from practicing the invention.⁷ One of the difficulties with the current state of the law is that the copyright law effectively grants a monopoly to a software owner without the public obtaining any disclosure. Due to mask work protection,⁸ or simply because a string of bits—the ones and zeroes of which software is made—is not readable, the public has no disclosure for inventions to be built upon. The mask work remains a trade secret, using that term in a layperson's sense, while an effective monopoly in favor of the author is created. Much of the case law and controversy is an explicit and implicit debate about the balance between the encouragement of disclosure and the grant of monopoly.

A recent article in April 1995 entitled *Patenting Mathematical Algorithms, Floppy Disks, and Other Computer Program-Related Subject Matter*⁹ (presented at the Computer Law Association Annual Law Update) proposes that software patents be allowed solely on the basis of utility and advanced the jarring proposition that formulae, including $E=mc^2$, be patentable. This author thought the proposition that $E=mc^2$ be patentable was so obscene that he was jarred into commencing this article. Shortly thereafter, the PTO published its Request for Comments on Proposed Examination Guidelines for Computer-Implemented In-

4. D.C. Toedt, *Patenting Mathematical Algorithms, Floppy Disks, and Other Computer Program-Related Subject Matter*, THE 1995 COMPUTER & TELECOMMUNICATIONS LAW UPDATE COURSE MATERIALS (The Computer Law Assn. 1995) [hereinafter *Patenting Mathematical Algorithms*]; see also D.C. TOEDT III, THE LAW AND BUSINESS OF COMPUTER SOFTWARE (Clark Boardman Callaghan, Supp. 1995).

5. 413 U.S. 49, (1973) (Brennan, J., dissenting).

6. *Id.* at 92.

7. "The Congress shall have power . . . To promote the Progress of Science and useful Arts, by securing for limited Times to Authors and Inventors the exclusive Right to their respective Writings and Discoveries." U.S. CONST. art. I, § 8, cl. 8.

8. See 17 U.S.C. § 901 (1994).

9. *Patenting Mathematical Algorithms*, *supra* note 4.

ventions,¹⁰ to which an earlier version of this article was submitted in support of a comment. These guidelines were revised and are contained in the Manual of Patent Examining Procedure in the Chapter on Patentability.¹¹ More recent revisions have been made to the Examining Procedures.¹²

In the recent case of *In re Trovato*,¹³ the Court of Appeals for the Federal Circuit vacated its earlier decision rejecting a claim and “remanded [the case] for consideration in light of *In re Alappat* and any new guidelines adopted by the Patent and Trademark Office for examination of computer-implemented inventions.”¹⁴ A dissent to the vacatur complained that “[e]ven if new guidelines are adopted, they must yield to precedent from this court and the Supreme Court, i.e. the *law* on section 101.”¹⁵

These software patent guidelines for computer-implemented inventions seem to render unpatentable formulae, music, and the like embedded in software. The guidelines also happily clarify that data storage devices will be patentable. However, this article agrees with the *Trovato* dissent and suggests that the guidelines will not significantly narrow the morass of conflicting case law about software programs.

B. *Thesis and Summary of Proposed Test*

The major thesis presented in this article is a focused standard of software patentability, in particular for pure computational methods or algorithms directed to the manipulation of numbers operating on a computer. The general philosophy is to compel inventors to narrow their claims to an algorithm expressed in terms of its utility and then to require that the particular utility or functionality be expressed in the claim as a limit on the claim, thus precluding the patent monopoly from being overbroad. As a corollary, any person is free to use or perhaps to patent the algorithm for a different utility outside the claims. The standard is consistent with existing statutes and with the trend which can be seen in the case law favorable to software patents.

10. 60 Fed. Reg. 28,778 (proposed June 2, 1995) [hereinafter *Proposed Guidelines*].

11. PATENT AND TRADEMARK OFFICE, MANUAL OF PATENT EXAMINING PROCEDURE, § 2106, rev. 1 (1995) [hereinafter MPEP].

12. *Examination Guidelines for Computer-Related Inventions*, 51 PAT. TRADEMARK & COPYRIGHT J. (BNA) No. 1262, at 422 (Jan. 25, 1996) [hereinafter *Amended Guidelines*].

13. 60 F.3d 807 (Fed. Cir. 1995).

14. *Trovato*, 60 F.3d at 807 (remanding an earlier decision rejecting claims to a method for placing data in a data structure to solve a “shortest path problem,” and a machine claim combining “means for function” clauses to solve for a least cost path).

15. *Id.* at 808 (Nies, J., dissenting).

The standard provides that software be patentable, even if the invention effects only a transformation of numbers or data. The point of novelty in all claims must be set forth in terms of function directed to a particular object to be achieved, not in terms of general function or means plus function. A general restriction to a technological area is also insufficient.

Specifically, the claim of the application of the software algorithm must be narrowed to a defined set of utilities or applications. Those utilities must be expressed in terms of manipulation of a particular input of numbers directed to a technologically useful end.

Each of these criteria should be contained in the claim: (a) at least an implicit understanding of the particular type of input covered by the claim algorithm, or perhaps the actual description of input, (b) the manipulation or transformation that will occur, and (c) the object to be achieved from the transformation and its functionality that operates as a limitation set forth in the claim. Therefore, the claim must be substantially in the form of:

What is claimed is:

a method embedded in a set of computer-translatable instructions comprising:

[recite steps]

so that input is [transformed] in conjunction with a computer into [output of a specified character for a specified [technologically useful application.]]

A corollary to requiring a statement of a claim having the conceptual framework of (a) input, (b) transforming function, and (c) object of transformation is that it must be described in the English language, not in a programming language. This is important to deal with the interesting problem of expert systems that invent algorithms. A patent on a method that, based on certain input, invents algorithms for a specified purpose may be tolerable, but a patent on methods not able to be described in the English language, which methods are for unspecified purposes, should founder as likely too overbroad a grant of a monopoly.

If only a pure transformation of numbers is involved, the claim may not only have to be narrow in its language of function directed to an object or end result, but may also have to expressly state, in at least broad terms, the type of input to which the function and transformation is directed.

The software patent should describe and claim an invention that improves an already known or obvious way to accomplish the claimed

function. Put more generally, the algorithm cannot be preemptive of the only method to transform input to a certain output.

If a machine claim to a computational method operating on a computer is allowed, the only type of machine claim allowable with respect to the pure computational method will be a claim to a machine using that specific allowable method. Means plus function clauses as elements of a machine claim involving an algorithm or formula should generally not be allowable unless the clauses precisely echo the steps of an allowable method claim. Disguised methods buried in a machine claim to a computer-implemented invention are to be ferreted out and analyzed as methods.

This required conceptual framework of a claim will give the public the protection it needs from too broad a grant of a monopoly in favor of the inventor, while receiving the disclosure of the formula or steps, and will allow the inventor to practice the invention for a period of time and for a particular purpose.

C. General Outline of Article

The article begins with a general overview of patent policy and a selection of cases from the potpourri of decisions in the area of computer-implemented inventions. A critique of other tests as a tool to forge a new test is presented. The detailed test is presented and applied to a number of examples, from both ends of the patentability spectrum, that most courts would agree should and should not be patentable. This article continues by discussing the constitutional background and details a new standard. Judge Archer's dissent in *In re Alappat*¹⁶ is analyzed to test and shape the proposed standard. The article then discusses the PTO guidelines. Appropriate standards of patentability are examined for four paradigms: (a) pure computational methods, (b) claims to programmed apparatus, (c) claims to data structures, and (d) claims to program storage devices.¹⁷

16. *In re Alappat*, 33 F.3d 1526 (Fed. Cir. 1994) (en banc) (Archer, C.J., concurring in part and dissenting in part).

17. *See generally Patenting Mathematical Algorithms*, *supra* note 4, i–ii.

II. PATENT POLICY AND THE CONFUSION OF CURRENT LAW

A. *Constitutional Policy*

“The Patent Clause of the Constitution empowers the Congress to ‘promote the Progress of . . . useful Arts, by securing for limited Times to . . . Inventors the exclusive right to their . . . Discoveries.’”¹⁸

The Patent Clause of the U.S. Constitution and the public’s contract of monopoly granted to an inventor, after 200 years of interpretation, clearly exist for the benefit of the public—for “We the People.” The courts, which have the last word, along with the PTO have been appropriately cautious about newer technologies, particularly in terms of allowing patents. Such caution should be dictated by the sure knowledge that the framers of the Constitution had little understanding of the “laws of nature”¹⁹ and science as we do today and certainly knew nothing of the modern digital computer. Caution is important because we cannot predict precisely how dangerous the allowance of an exclusive right will be to limiting the advance of technology.

The aspirations of the previous paragraph can be translated into an expectation of the public in return for its generous reward of a monopoly. That public expectation has two basic components: first, the public expects to obtain disclosure so that the rest of the world can use what is disclosed in new and useful ways. Second, the public expects the next inventor, having reviewed the first inventor’s disclosure, to be encouraged to have a new, useful, and non-obvious improvement. The public then expects this inventor to apply for a patent and to disclose that new, useful, non-obvious improvement, so that the cycle of invention will continue. The patent monopoly, very simply, is a carrot to induce disclosure.

So-called natural laws, for which the courts have uniformly denied patent protection,²⁰ easily meet the first component of public expectation because they are useful and new and because the public certainly desires disclosure. However, they founder on the second component of public expectation. There is no possible improvement to the natural law that the next inventor can deliver for the public benefit. The natural law is preemptive, and it stands alone as an unimprovable concept.

18. *In re Alappat*, 33 F.3d 1526, 1552 (Fed. Cir. 1994) (quoting U.S. CONST. art. I, § 8, cl. 8).

19. *Diamond v. Diehr*, 450 U.S. 175, 185 (1981) (explaining that the Court has found such things to be excluded from patent protection).

20. *Id.*

The Supreme Court recognized this in a different way in *Graham v. John Deere Co.*,²¹ when it observed that non-obviousness “draw[s] a line between the things which are worth to the public the embarrassment of an exclusive patent, and those which are not.”²² From a natural law, there will be no expected gain; thus, there is no policy reason for having to descend to the disgrace of granting a monopoly. As a corollary, if the monopoly grant is too broad, for policy reasons we simply determine that the broad monopoly is not worth the disclosure and do not allow a patent.

The Supreme Court articulated this concern in the area of software patents by observing in *Gottschalk v. Benson*²³ that “[t]he 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.”²⁴ The facts of *Benson* involved a claim to an algorithm for converting decimal numbers into binary numbers. Granting a patent on a natural law (e.g., $E=mc^2$, $v=\lambda f$, or Maxwell’s equations), regardless of whether it is encoded in a software program, would prevent patenting future improvements based on that natural law.

However, the dilemma is still not resolved. We have discerned that we do not desire patents on mathematical formulae containing a law of nature, a scientific truth, or an abstract idea;²⁵ we are comfortable with existing tests in which formulae tied to structure are patentable. Yet, what do we do about a terribly useful computer algorithm called a “bubble sort,” discussed later in this article, which is uniquely useful on a computer?

B. Supreme Court Cases

The usual starting point in Supreme Court jurisprudence for software patents is *Gottschalk v. Benson*.²⁶ In that case, “the Supreme Court held that claims to a method of converting binary-coded decimal numbers into pure decimal numbers did not recite an invention or discovery within §101, and thus were ineligible for patent protection.”²⁷

21. 383 U.S. 1 (1966).

22. *Id.* at 9 (quoting BASIC WRITINGS OF THOMAS JEFFERSON 713 (Philip S. Foner ed., 1944)).

23. 409 U.S. 63 (1972).

24. *Id.* at 71–72.

25. *See Diamond*, 450 U.S. at 185–86.

26. 409 U.S. 63 (1972).

27. *In re Alappat*, 33 F.3d 1526, 1555 (Fed. Cir. 1994) (en banc) (Archer, C.J., concurring in part and dissenting in part) (referring to *Gottschalk v. Benson*, 409 U.S. 63 (1972)).

The author submits that the *Benson* case must be viewed on its facts, which presented serious novelty questions and extremely broad claims. The *Benson* Court was extremely troubled that the formula “sought to be patented was a mathematical formula that had no substantial practical application except in connection with a digital computer, . . . [and] the patent would wholly pre-empt the mathematical formula and in practical effect would be a patent on the algorithm itself.”²⁸

The Supreme Court did infer that the patent law would need to “be extended [i.e., amended] to cover these programs.”²⁹ If that statement is broadly viewed, it would once and for all eliminate software patents. As subsequent cases would illustrate, the Court was not reaching that far, but the Court’s discomfort was and still is apparent.

The *Benson* case also made it clear that if a computer program, as a general matter, is deemed to be a mathematical formula, then either computer programs and software will not be patentable, or we will have to resort to the fiction of declaring a software algorithm not to be a formula. In sum, *Benson* has to be thought to be narrowed by technology and later case law.³⁰

By 1978, the Supreme Court in *Parker v. Flook*³¹ was obviously struggling to define patentability for what was clearly intended to be a computerized process for updating an alarm limit which signaled an abnormality in a catalytic chemical process.³² The Court observed that there was no significant limitation on the use of the method in the claims and denied the inventor his patent. The Court was also troubled that:

[t]he chemical processes involved in catalytic conversion of hydrocarbons are *well known*, as are the practice of monitoring the chemical process variables, the use of alarm limits to trigger alarms, the notion that alarm limit values must be recomputed and readjusted, and the use of computers for “automatic monitoring-alarming.” (emphasis added)³³

A known method embodied in software should not be patentable under any standard, including that set forth in this article. In *Flook*, the

28. *Benson*, 409 U.S. at 71–72.

29. *Id.* at 72.

30. Another shibboleth to invalidate software patents which appears shelved is the “mental processes” doctrine. Pamela Samuelson, *Revisited: The Case Against Patent Protection for Algorithms and Other Computer Program-Related Inventions*, 39 EMORY L.J. 1025, 1043 (1990).

31. 437 U.S. 584 (1978).

32. *Id.* at 585–86.

33. *Id.* at 594 (footnote omitted).

Court rejected the “notion that post-solution activity, . . . can transform an unpatentable principle into a patentable process”³⁴

By the time of *Diamond v. Diehr*,³⁵ the Supreme Court conceded that integration of a digital computer and software programs with “post-solution activity” would render an invention eligible for a patent.³⁶ The facts of that case involved the integration of the Arrhenius equation with a method of operating a rubber molding machine while monitoring the cure of a rubber product being processed.³⁷ In the recent case of *In re Alappat*,³⁸ Chief Judge Archer summarized the conclusions of the Supreme Court in *Benson* and *Diehr* as follows:

Under [these cases] the posing and solution of a mathematical *Flook* function is non-statutory subject matter. It is non-statutory even if the particular mathematics is limited to performance in digital electronic circuitry or a general purpose digital computer, even if the mathematical operations are alleged generally to have some application in one or various technologies, and even if the solution of the function is said generally to “represent” something of physical or technologic relevance. On the other hand, an invention or discovery of a process or product in which a mathematical operation is practically *applied* may be statutory subject matter. The fact that one element of the claimed process or product is a programmed digital computer or digital electronics performing a mathematical function does not necessarily preclude patent protection for the process or product. In this way, the door remains open to the advancement of technologies by the incorporation of digital electronics. But the mere association of digital electronics or a general purpose digital computer with a newly discovered mathematical operation does not per se bring that mathematical operation within the patent law.³⁹

C. The Federal Circuit

The Federal Circuit’s majority conclusion in *In re Alappat* and the conventional wisdom have been that so long as there is significant post-solution activity or sufficient physical structure associated with a

34. *Id.* at 590.

35. 450 U.S. 175 (1981).

36. *Id.* at 191–93.

37. *Id.* at 177–81.

38. *In re Alappat*, 33 F.3d 1526, 1560 (Fed. Cir. 1994) (en banc).

39. *Id.* at 1557 (Archer, C.J., concurring in part and dissenting in part).

software claim, the Supreme Court would likely allow a patent on the proposed invention.⁴⁰

As evidenced by the fifty-one page dissent by Chief Judge Archer in *In re Alappat*, the Federal Circuit has not settled comfortably to any conclusions in the software patent area. *In re Alappat* was an en banc climax to a difficult decade of cases.

Prior to *In re Alappat*, the test for patentability of computer-implemented inventions was referred to as the *Freeman-Walter-Abele* test. The test was and is too narrow to allow any significant patent, and arguably any patent at all, on traditional software.⁴¹ In essence, the *Freeman-Walter-Abele* test is (a) “whether the claim directly or indirectly recites an ‘algorithm’ in the *Benson* sense of that term,”⁴² (b) “whether in its entirety it wholly preempts that algorithm,”⁴³ (c) whether the algorithm is “applied in any manner to physical elements or process steps,”⁴⁴ and (d) whether the invention’s “application is circumscribed by more than a field of use limitation or non-essential post-solution activity.”⁴⁵ The *Freeman* component of the test is whether the claim recites an algorithm and whether it preempts the algorithm.⁴⁶ This part of the test is useful for the proposed test with some clarification. The *Walter* component of the test requires application to physical elements or process steps, but rejects a field of use limitation as a useful analytical tool.⁴⁷ By requiring a “physical aspect” or application to statutorily allowable process steps (implying a requisite physical transformation), the *Walter* component renders unpatentable the pure transformation of numbers such as sorting. The *Abele* component of the test softens the *Walter* component by providing that if the claim presents statutory subject matter without the algorithm, it likewise is statutory with the algorithm.⁴⁸ Unfortunately, reference is made in the context of the phrase “[must be] otherwise statutory,” (e.g., statutory subject matter without the algorithm) to the phrase: “albeit inoperative or less useful without

40. *Id.* at 1560 (Archer, C.J., concurring in part and dissenting in part); see also *Patenting Mathematical Algorithms*, *supra* note 4; *Proposed Guidelines*, *supra* note 10, at 28,780; MPEP, *supra* note 11, § 2106; *Amended Guidelines*, *supra* note 12, at IV.B.2.e.

41. The test is described in both *Patenting Mathematical Algorithms*, *supra* note 4, § 6.03, and in an article of the eminent Professor Donald S. Chisum, *The Patentability of Algorithms*, 47 U. PITT. L. REV. 959, 1003 (1986). See also *In re Freeman*, 573 F.2d 1237 (C.C.P.A. 1978); *In re Walter*, 618 F.2d 758 (C.C.P.A. 1980); *In re Abele*, 684 F.2d 902 (C.C.P.A. 1982).

42. *Freeman*, 573 F.2d at 1245; see also Chisum, *supra* note 41, at 1002–03.

43. *Id.* at 1245; see also Chisum, *supra* note 41, at 1002–03.

44. *Walter*, 618 F.2d at 767; see also Chisum, *supra* note 41, at 1003.

45. *Abele*, 684 F.2d at 907; see also Chisum, *supra* note 41, at 1003.

46. Chisum, *supra* note 41, at 1002–03.

47. *Id.* at 1003.

48. *Id.*

the algorithm.”⁴⁹ In order for the claim to be “otherwise statutory,” it must have some physical aspect. The requirement of a physical aspect effectively eliminates patents for inventions which only consist of the pure transformation of numbers.⁵⁰ It is hard to see how a test against “inoperative or less useful” statutory subject matter will be productive; thus, a different direction is taken in this article to focus on transformation directed to a functional object.

In the majority opinion of *In re Alappat*, the Federal Circuit, sitting *en banc*, characterized the invention as relating “generally to a means for creating a smooth waveform display in a digital oscilloscope.”⁵¹ Based on the statutory language of 35 U.S.C. §112(6), the Federal Circuit held that means plus function elements had sufficient physical structure inherent in an electronic machine to be patentable:

Although many, or arguably even all, of the means elements recited in claim 15 represent circuitry elements that perform mathematical calculations, which is essentially true of all digital electrical circuits, the claimed invention as a whole is directed to a combination of interrelated elements which combine to form a machine for converting discrete waveform data samples into anti-aliased pixel illumination intensity data to be displayed on a display means. This is not a disembodied mathematical concept which may be characterized as an “abstract idea,” but rather a specific machine to produce a useful, concrete, and tangible result.⁵²

The vacated holding in *In re Trovato* contained a few hints on how at least one panel of the Federal Circuit felt about software patents.⁵³ The invention at issue, in substance, was a patent on a solution method to a critical path problem or “shortest path problem.” A method for storing data in order to enable a solution was claimed as well as a method disguised as a machine which, in a series of means for function elements, claimed a “means for identifying a least cost path between two positions in the discretized representation based on the respective costs.”⁵⁴ The *Trovato* court correctly concluded that the claims were to a mathematical algorithm. The only physical step seemed to be a computer readout. Yet, the *Trovato* court seemed to leave open, however slightly, a window of opportunity for a patent: “Without further

49. *Id.*

50. *Abele*, 684 F.2d at 907.

51. *In re Alappat*, 33 F.3d 1526, 1537 (Fed. Cir. 1994) (en banc).

52. *Id.* at 1544 (footnotes omitted).

53. *In re Trovato*, 42 F.3d 1376 (Fed. Cir. 1994), *vacated*, 60 F.3d 807 (1995).

54. *Id.* at 1378.

application or connection to a technical art, we cannot say that Trovato's claims pass muster under the alternative analysis of statutory subject matter expressed in *Warmerdam*.⁵⁵

The facts of *Warmerdam* involved a bubble hierarchy scheme to direct a robot. A claim was allowed. The alternative analysis seemed to involve determining whether the claim was merely to an abstract idea.⁵⁶

One of the reasons a new approach and test is proposed for software patents is that the requirement of "physical structure" is perverse and has been perverted, notably reaching a climax in the recently proposed PTO Guidelines for Computer-Implemented Inventions.⁵⁷ Essentially, the Proposed Guidelines and the *In re Alappat* majority conclude that if the Supreme Court wants structure, "we'll give them structure": i.e., "means plus function" elements will be sufficient structure to allow a patent on an algorithm. Requiring inventors to call an apple an orange, that is to call the algorithm embedded in software a "means plus function" instead of a "method step," is a venture on the wrong track which will grant a monopoly without encouraging disclosure. Moreover, "means plus function" clauses tend to be broad. An important objective of allowing any claims to algorithms embedded in computer programs is to limit their breadth and, in particular, not to encompass future undiscovered "means plus function" inventions. Such limitation is necessary in order to avoid the preemption issue rightly raised in *Benson*.⁵⁸

III. DEVELOPMENT OF A NEW TEST BASED ON SAMPLE ENGINEERING PROBLEMS AND BASED ON SEMINAL CASES IN THE PATENT LAW

A. *A Spectrum of Examples Illustrating Why a Pure Utility Test Is Inadequate*

An understandable example of a formula that crosses the spectrum from pure unpatentable formula to a formula easily contained in a patent is the basic wave speed calculation algorithm, $v = \lambda f$. The meaning of this mathematical formula is that the speed of a wave is equal to its wavelength λ times its frequency f . This formula pointedly illustrates why a pure utility test, as proposed in *Patenting Mathematical Algorithms*,⁵⁹ is not socially desirable or consistent with U.S. patent policy.

55. *Id.* at 1381 (citing *In re Warmerdam*, 33 F.3d 1354 (Fed Cir. 1994)).

56. *Warmerdam*, 33 F.3d at 1359.

57. *Proposed Guidelines*, *supra* note 10. See also MPEP, *supra* note 11; *Amended Guidelines*, *supra* note 12.

58. See *Gottschalk v. Benson*, 409 U.S. 63, 71-72 (1972).

59. *Patenting Mathematical Algorithms*, *supra* note 4.

Posit the allowance of the following claim to calculate wavelength:

What is claimed is:

1. A method of calculation of the speed v of a wave calculated according to the formula, $v=\lambda f$ where λ is the wavelength and f is the frequency.

Assume the claim is supported by a disclosure in the context of measuring the wavelength of sound. The claim and disclosure clearly have utility. Under the proposed “utility-only” standard of *Patenting Mathematical Algorithms*, this formula, when discovered, upon suitable application, should have been patentable.

The inventor who obtained a patent including this claim would have discovered that he can always know the wavelength of sound if he measures the pitch (a musician’s term for the frequency) because the inventor “knows” by physical measurement that the speed of sound is approximately 110 meters per second.

What the inventor did not know is that the speed of sound in a particular medium varies. What neither the patent office nor the inventor knew was that the formula also applied to the speed of light. Once this was discovered, the inventor would be delighted with the breadth of his or her monopoly.

The following week, Einstein realizes that this is a great formula. He theorizes that the speed of light is always constant at a velocity c . He decides to use the formula to determine the smallest opening that a camera lens may have for a given frequency of light in order to take pictures without the Fresnel effect.⁶⁰ He proposes to use the formula $v=\lambda f$ in the following way: for a speed of light v and for a frequency of light f , which he would have managed to measure, he proposes to determine the wavelength of light λ according to the formula $v=\lambda f$. Then, he would propose to apply that calculation of λ to determine the smallest opening (what we now call the “f stop”) on a camera lens for a given wavelength. There is no question that Einstein would have wanted to produce such a camera, given the size of the camera market. Most importantly, and in rebuttal to the proposition that such pure formulae should be patentable solely based on their utility, there is no question that Einstein would have needed to use the formula $v=\lambda f$.

60. The Fresnel effect in a camera lens can be described as follows: If light is passed through a sufficiently small opening, it will diffract, causing the appearance of alternating bands of black and the color. Because a camera lens is circular, these appear on a photograph as rings and distort the photograph.

In this hypothetical, he would likely have infringed the formula claim set forth above, if that claim were allowable.

There is also no question that the claim in a patent for the camera built with an opening determined according to the formula $v=\lambda f$ would be patentable under existing case law. We feel good about this patent. It is the opposite extreme of a formula patent. It has that traditional look and feel of “structure.”

The previously referenced article entitled *Patenting Mathematical Algorithms*, after discussing the unsteady course of jurisprudence in the software patent area, proposes that utility be the sole test for statutory subject matter. The most jarring proposition advanced is:

Under a *Chakrabarty*-inspired statutory utility analysis, Einstein’s equation $E=mc^2$ or the Pythagorean theorem $a^2+b^2=c^2$ might have been potentially patentable, notwithstanding contrary dicta in *Flook* and *Chakrabarty* and Chief Judge Archer’s concern in his *Alappat* dissent. Each of these equations is simply a convenient notational representation—no less and no more—of a computational method, a human-created mathematical modeling process, that includes data-gathering and multiplication steps to generate an approximation of a real-world natural phenomenon. Likewise, the familiar equation $NaCl \Leftrightarrow Na^+ + Cl^-$ serves as a shorthand expression of a mathematical modeling process for predicting sodium-[sic] and chloride ionization in an aqueous salt solution.

Einstein’s equation, the Pythagorean theorem, and the sodium chloride [sic] ionization equation cannot properly be considered “laws of nature” [and should therefore be patentable]

The human origin and imperfect nature of such equations is highlighted by the fact that their accuracy is always an issue. As a well-known example, the modeling process expressed notationally as Newton’s second law of motion, $F=m(dx^2/dt)$ or more commonly $F=ma$, has proved to be unacceptable for predicting or explaining relativistic or quantum phenomena. *Flook* was therefore wrong in asserting in dicta that “a scientific principle, such as that expressed in [*Flook*’s] algorithm, reveals a relationship that has already existed.”⁶¹

The author of this article therefore disagrees that utility should be the sole test. The formula $v=\lambda f$ has near universal application, and, in contrast to the *Patenting Mathematical Algorithms* reference that formulae that are natural laws function only as an approximation, this

61. *Patenting Mathematical Algorithms*, *supra* note 4, § 6.04 (footnotes omitted).

particular formula appears to have application in all known fields of technology, including quantum mechanics.⁶² The argument of approximation as a justification for a pure utility test would therefore founder.

*B. The “Bubble Sort” and the Critical Requirement
of a “Transformation”*

Turning to a pure software example, the “bubble sort” is an example of pure manipulation of numbers that is understandable in one reading of this article and is an algorithm that the author believes would have deserved patent protection. The reason one can feel comfort about patenting the bubble sort algorithm is that it transforms a set of numbers into a different set of numbers in a useful way.

Suppose you have a list of n numbers and you are directed to sort it in ascending order. Most of us search the list and look for the smallest number and write that down on a separate list. We then look at the list and search for the next largest number. We repeat this until we are at the end of the first list.

Put another way, we pick a number, see if it is the smallest compared to the list, and reject it back to the list or select it as the new “smallest.” Once we see it is the smallest, we put it on a separate list and start over.

One can compute that for a list of n numbers—you have to look at n numbers the first run, $n-1$ the second run (because the result from the first search is removed), $n-2$ the third run, and so on until one number is left which must be the highest remaining. As it turns out, $n(n-1)/2$ searches are required. This means that this method of sorting consumes a number of comparisons in proportion to n^2 .

In the middle 1960’s, a creative person devised a method which this author learned as “bubble sorting.” Bubble sorting called for all numbers to be placed in a triangularly shaped set of nodes which we called a tree, with the point, or top, of the tree placed arbitrarily at the top of the page. A number was placed at the upper point of the tree, two numbers were placed in the first level down and “linked” to the top number, four numbers were placed in the second level in two “sets of two” “linked” respectively to the two numbers above, and so on until the list of numbers to be sorted was exhausted. A rough figure of the first three levels is set forth below, with the circles corresponding to nodes in which numbers would be written and in which the lines constitute links with the “next bubble up.”

62. The universal application is subject only to difficulty of measurement and verification at a quantum mechanic level because of Heisenberg’s uncertainty principle.

```

      0
     /\
    00
   /\ /\
  0000

```

As we all know, few lists to be sorted are completely out of order. Usually, there are at least a few low numbers in front of some higher numbers on the list, simply because of random distribution. The bubble sort takes advantage of this phenomenon. The basic principle is that starting at the bottom right corner, if a number is lower than the leftmost number in its own triad, it is exchanged with that number. Then, the leftmost number is compared to the top node in the triad. If the leftmost number is lower than the top number in the triad, the numbers are exchanged, and the algorithm is worked backward through the triad and any lower levels, but such working backwards is only done if an exchange of numbers at a higher level occurs. For a list of n numbers in base 10, the sort can be shown to function at a magnitude of $n \times \log_{10} n$ comparisons.

The relative advantage of the bubble sort may not seem large if n is 100, for it is the difference between 200 and $9900/2$. But make that list one million numbers, and the difference in magnitude is the difference between six million and one trillion/2 comparisons.

Now that's an improvement! It is the pure manipulation of numbers. It is not particularly useful to employ this algorithm by hand because it is awkward mechanically; you have to erase too many times as you "bubble" the numbers up and down. This author submits that if a person invented a software program to do that and wrote the claim in such a way that the monopoly is not too broad, "We the People" would very much like to have that disclosure. Under the standard to be proposed in this article, when someone could figure out how to sort for a purpose not claimed by the inventor, or how to sort by using different steps than those claimed by the inventor (in fact, there are several different sets of steps), that person too could have a patent.

The concept of transformation is key to resolution of a feeling of comfort about software patents and determination of a practical standard. "Transformation and reduction of an article 'to a different state or thing' is the clue to the patentability of a process claim that does not include particular machines."⁶³ The reason one can feel comfort about

63. *Gottschalk v. Benson*, 409 U.S. 63, 70 (1972); *see also* *Diamond v. Diehr*, 450 U.S. 175 (1981).

the bubble sort algorithm is that it transforms a set of numbers into a different set of numbers in a useful way.

Transformation is defined in terms of the word “transform”: “*v.t.* 1. to change in form, appearance, or structure; metamorphose. 2. to change in condition, nature, or character; convert . . . *v.i.* 6. to undergo a change in form, appearance, or character; become transformed”⁶⁴

Courts are now comfortable with a patent that employs pure natural laws or formulae to determine how elements may be structured into an invention,—e.g., the camera example above using the formula $v=\lambda f$ to determine the smallest possible lens opening.⁶⁵ The author, and many others, would be comfortable with a software patent on this pure manipulation of numbers, despite the author’s obvious hostility to a patent on the mathematical formulae $v=\lambda f$ or $E=mc^2$.

As will be seen however, transformation plus pure utility cannot be the only test. The foundation for why it may not be the only test lies in constitutional authorization and policy. The solution will lie in the concept of transformation plus utility directed to an objective.

C. Matrices and the Requirement of Transformation Plus Utility

1. Mere Transformation Plus Utility Is Not Sufficiently Narrow for Software Patents

A review of matrix algebra illustrates why a test having only the requirements of utility and transformation of a data set will still raise the hurdle too high for future disclosure and will not achieve the second component of public expectation. Another factor will be required in any practical standard.

Matrices are wonderful objects that can achieve remarkable efficiencies. They are especially useful in several classes of problems generally referred to as optimization problems, requirements problems, Markov processes with multiple states, and transportation problems. A classic example of a network problem is determining the shortest or “critical” path through a series of steps which may require certain prerequisites before performance. A construction project of a building or a spacecraft is a good example of a network problem. Remanufacturing or programming projects are another example. A transportation problem is the problem of how to move a required number of goods or services

64. THE RANDOM HOUSE COLLEGE DICTIONARY 1395 (rev. ed. 1982).

65. See generally *Diamond v. Diehr*, 450 U.S. 175 (1981) (discussing when a mathematical formula can validly be included in a patent claim for a rubber curing process).

over a series of given routes in an optimal way between a given set of origins and destinations.

A matrix is a table of numbers. Assume it has m rows and n columns. It may be multiplied by any matrix that has n rows. Assume the second matrix has p columns. The matrix yielded on the multiplication of the first matrix A (an $m \times n$ matrix) by matrix B (an $n \times p$ matrix) is a matrix C (an $m \times p$ matrix). Assume rows are indexed and referred to by an index “ i ” and columns are indexed and referred to by an index “ j .” A particular element of matrix A , for instance in the “ i ’th row” and in the “ j ’th” column, is thus referred to as a_{ij} . A particular element of matrix B in the “ i ’th row” and in the “ j ’th column” is referred to as b_{ij} .

$$\begin{bmatrix} a_{1,1} & a_{1,2} \\ a_{2,1} & a_{2,2} \\ a_{3,1} & a_{3,2} \end{bmatrix} \times \begin{bmatrix} b_{1,1} & b_{1,2} & b_{1,3} \\ b_{2,1} & b_{2,2} & b_{2,3} \end{bmatrix} = \begin{bmatrix} c_{1,1} & c_{1,2} & c_{1,3} \\ c_{2,1} & c_{2,2} & c_{2,3} \\ c_{3,1} & c_{3,2} & c_{3,3} \end{bmatrix}$$

In the above example, a 3×2 matrix multiplied by a 2×3 matrix yields a 3×3 matrix.

Indexing rows on an index “ i ” and columns on an index “ j ,” any particular element C_{ij} of matrix C will be

$$C_{i,j} = \sum_{k=1}^n A_{ik} B_{kj}$$

The layman’s way of doing this is as follows: the elements of the first row of matrix A are set next to the elements of the first column of matrix B , each set of juxtaposed elements is multiplied together, the products are added up, and the sum is the new element in the first row and first column of the new matrix C .

Even as first symbolized and invented, this formula should not be protected in a software patent. It clearly transforms a data set (just as a bubble sort transforms a data set). It can be done by hand or faster by computer. Yet, to allow this algorithm to be patentable would raise a hurdle to the entire science of operations research for an extended period of time. Optimal solutions in operations research problems attempt to select an extreme point in a multidimensional analysis on the so-called “polyhedral cone” by using matrix algebra, including matrix multiplication.⁶⁶ A patent on the matrix algebra of multiplication because it satisfies the requirements of utility and transformation would

66. See generally HARVEY M. WAGNER, PRINCIPLES OF OPERATIONS RESEARCH, §§ 3.4–3.6 (Prentice-Hall, Inc. 2d ed. 1975) (discussing polyhedral cones and their use in solving linear optimization problems).

close off technological development in too broad a fashion. Embodying the formula in a machine, i.e., embedding it in hardware or embedding it in software, would not resolve the public's concerns. Any standard must thus involve more than the mere requirement of transformation.

One solution to consider as a requirement for a patent would be the concept of a "short-cut," an improved method over an old method. Let us suppose a software program written in a novel way enables the matrix multiplication to be short-cut. In fact, there exist just such short-cuts in matrix multiplication which are used to solve particular kinds of network and transportation problems.

Unfortunately, the concept of allowing a patent because an algorithm is a short-cut is overbroad. For instance, to allow a patent on a method of solving any and all network or transportation problems merely because the newly-invented method is a short-cut would be overbroad. The fact that the problem could be solved in a lengthier manner is still not satisfactory. In other words, the proposed patent for the short-cut is not preemptive. There still must be a further limitation on the patent monopoly to avoid an overbroad reach.

For a specific example, if Dantzig's simplex algorithm,⁶⁷ known as the Simplex Method of solving linear programming problems, had been patented without limit because it had utility and effected a transformation, the science of operations research would have been severely impeded. Although Dantzig's simplex algorithm was not the only means of solution of operations research problems, it was a substantially improved method of Gaussian elimination⁶⁸ which eliminated many trial and error steps in a large problem.⁶⁹ It is very troublesome that embedding these algorithms in software that abstractly transforms numbers would bar the rest of the world from using them, absent paying a monopoly price. The same argument about monopoly price, of course, has been made concerning patents generally, and a more humanly-compelling argument can be made about medical devices. In contrast to medical devices, however, the author submits that the broad and unanticipated uses of formulae and algorithms raise issues of wider concern because the purposes for which formulae or algorithms can be used are infinite, and usually the universe of possible uses is unknown at the outset of the patent process. It is even more troubling if there is no compulsory licensing because large fields of a basic building block can

67. *Id.* § 4.4.

68. *Id.* § 4.2.

69. Dantzig's algorithm offered a far more rapid convergence on a solution. This algorithm is still important in the advanced world of incredibly fast, massive supercomputers. *See id.* §§ 4.2–4.3.

be arbitrarily blocked off. Medical devices have a more limited universe of applications and realistically present fewer concerns in this area. The solution to the troublesome concern of allowing patents on broadly useful algorithms embedded in software requires close attention to narrowing the claims to a specified function or list of functions that are technologically useful. It also will require courts to focus closely on abuse of monopoly power if a software program, as claimed, turns out to have broad reach and if the proposed license fees are disproportionate to the research and development costs. Note that the production costs of software, which are largely limited to copying and distribution, are almost irrelevant to the price of software.

2. Transformation Directed to a Utilitarian Object and Using that Object as a Limitation on the Scope of Claim

In the last century, an important advance in medical science was the use of sulfuric ether as an anesthetic. In the case in which the patent on that use was held invalid, the court stated important principles which are applicable to and translatable to the software patent area.⁷⁰

The *Morton* court held, “[t]he new force or principle brought to light must be embodied and set to work, and can be patented only in connection or combination with the means by which, or the medium through which, it operates.”⁷¹

The second clause of this holding which reads, “only in connection or in combination with means by which, or the medium through which, it operates,”⁷² should be a necessary element to a software patent, i.e., the claim must specify the method is implemented on a computer (analog or digital), that is the medium through which software operates.

The first clause from the *Morton* quotation reads, “the new force or principle brought to light must be embodied and set to work.”⁷³ In this clause lies an important and useful element of test for software patentability.

It is not mere utility that makes the claim and invention patentable; software is useful as are formulae, and this statutory requirement must continue. It is not a limitation to method-of-transformation claims only;

70. *Morton v. New York Eye Infirmary*, 17 F. Cas. 879 (C.C.S.D.N.Y. 1862) (No. 9865).

71. *Id.* at 884. The court also noted that a method must act on something “material,” that narrowness of vision in the computer age is the vise with which the courts are now grappling. Information is power and advances in the ability to manipulate numbers and information really are advances in technology that are useful.

72. *Morton*, 17 F. Cas. at 884.

73. *Id.*

a method or process is implicit in every algorithm. This should be a necessary recitation of any claim.

It is most important to narrow the claim to how it is “embodied and set to work” as the *Morton* court required. The claim of a formula or algorithm must be narrowed from the application—the transformation effected—of the software algorithm to a defined set of utilities or applications. Those utilities should be in terms of manipulation of a particular input of numbers directed to a technologically useful end. Each of these criteria should be contained in the claim: (a) at least an implicit understanding of the particular type of input covered by the claim algorithm (or perhaps the actual description of input), (b) the manipulation or transformation that will occur, and (c) the object to be achieved from the transformation and its functionality that operate as a limitation set forth in the claim.

One can speculate that the reason the Supreme Court finally accepted patents on algorithms is that the effect of requiring post-solution activity or structure is to necessarily limit the claim to “an object to be achieved from the transformation and its functionality that operate as a limitation set forth in the claim” as proposed above. For example, the claim in *Diehr* which used the Arrhenius equation was limited to a rubber curing process in a particular way; the claim was self-limiting, even with the abstract formula contained in the claim.⁷⁴ The test in this article is intended to be consistent with and better express the aspiration of the courts for a limited patent in a new area of technology.

3. Other Necessary Aspects: English Language Description of Method, Lack of Preemption by the Algorithm, and Demonstrable Reduction to Practice

a. Setting Forth the Method in the English Language

A corollary to these required criteria is that the claim must be described in the English language, not in a programming language. This is important to deal with the interesting problem of expert systems that invent algorithms. A patent on a method that, based on certain input, invents algorithms for a specified purpose may be tolerable, but a patent on the undescribed methods for unspecified purposes should founder because it is too overbroad a grant of a monopoly. It is also important to

74. See *Diamond v. Diehr*, 450 U.S. 175, 177–80, 187 (1981) (making clear that the monopoly on a mathematical formula extends only to its use in the computer-controlled process).

address the issue of a new short-cut method which will be discussed below.

This required conceptual framework of a claim will give the public the protection it needs from too broad a grant of a monopoly in favor of the inventor, while receiving the disclosure of the formula or steps, and will allow the inventor a period of time for a particular purpose to practice his or her invention.

b. The Algorithm Must Not “Pre-empt” the Field; It Cannot Be the Only Method of Accomplishing the Result Nor a Description of a Previously Unknown Relationship or Principle

Another important standard that should be met to prevent an overbroad monopoly is in the area of preemption. The software patent should describe and claim an invention that improves an already known or obvious way to accomplish the claimed function. To return to the bubble sort example, it is not the sorting that should be patentable—it is the new and useful mechanism of sorting. The Supreme Court in *Parker v. Flook*⁷⁵ realized this in a different way by observing that “a scientific principle, such as that expressed in [Flook’s] algorithm, reveals a relationship that has already existed.”⁷⁶ If there were no other way to sort numbers that were known, the bubble sort should not be patentable; it would, if there were no other way to sort, not reveal what we already know can be done. The same is true for matrix multiplication. If a pure computational method invention does not reveal a relationship that already exists, it is too basic, and the hurdle for the next inventor’s disclosure will be too high; e.g., Einstein will not be able to use $c=\lambda f$ in his camera patent if $v=\lambda f$ is already patented. However, a disclaimer to the basic formula in a patent containing a new revelation directed to an object should be acceptable.

Turning to the *Benson* case, the Court concluded that “[t]he method sought to be patented varies the ordinary arithmetic steps a human would use by changing the order of the steps, changing the symbolism for writing the multiplier used in some steps, and by taking subtotals after each successive operation.”⁷⁷ The Court concluded that was very troubled that “[t]he 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 pat-

75. *Parker v. Flook*, 437 U.S. 584 (1978).

76. *Flook* at 593 n.15 (1978).

77. *Gottschalk v. Benson*, 409 U.S. 63, 67 (1972).

ent would wholly pre-empt the mathematical formula and in practical effect would be a patent on the algorithm itself.”⁷⁸

Returning to the example of $v=\lambda f$, this preemption of the formula is why it should only be claimed in the context of a function to transform certain input into certain output. “If there is to be invention from such a discovery, it must come from the application of the law of nature to a new and useful end.”⁷⁹ This also overcomes the criticism in *Benson* that the “‘process’ claim is so abstract and sweeping as to cover both known and unknown uses of the BCD [binary-coded decimal] to pure binary conversion.”⁸⁰ The Court noted a broad variety of functions covered.⁸¹

The mere combination of a formula with a digital computer should be inadequate because it fails to satisfy the concept of novelty, not because there is anything intrinsically unpatentable about the combination with a digital computer. Based on Justice Douglas’ conclusion in *Benson v. Gottschalk* that the method “varie[d] ordinary arithmetic steps a human would use by changing the order of the steps,”⁸² the patent should fail on that basis alone.

To be intellectually honest, a computer program always uses a series of steps, i.e., a method, as Professor Allen Newell (a computer scientist, a non-lawyer, and non-patent practitioner) observed.⁸³ Those steps are usually a program, i.e., software directing a computer. Newell defines an algorithm as, “an unambiguous specification of a conditional sequence of steps or operations for solving a class of problems.”⁸⁴ This is inherently a process or method in the author’s view, and all software claims must be first analyzed in that context.

c. There Must Be a Demonstrable Reduction to Practice

Unlike traditional patent law, which usually does not require a working copy of the invention but only requires that one *can* be created, there should be a requirement of availability of a working copy—an actual reduction to practice—unless a step in the working copy (or more likely a program module) can be described and is disclosed, cited, and known in the art. In that case, only the novel steps need be disclosed, and a depository copy of that portion need be delivered to the PTO. This

78. *Benson*, 409 U.S. at 71–72.

79. *Id.* at 67 (citing *Funk Bros. Seed Co. v. Kalo Co.*, 333 U.S. 127, 130 (1948)).

80. *Benson*, 409 U.S. at 68.

81. *Id.*

82. *Id.* at 67.

83. Allen Newell, *Response: The Models are Broken, the Models are Broken!*, 47 U. PITT. L. REV. 1023, 1024–25 (1986).

84. *Id.* at 1024.

is key in order to limit software patents to patents on which new discoveries and inventions may be built.

The requirement of the test for a depository copy is not to restrict claims to a particular programming language nor to allow the next patent based on an artificial distinction of a new programming language for an old method. This is another reason for an English language disclosure of the method or modules to the method. Professor Newell correctly argues that an algorithm is broader than a computer program; an algorithm may be programmed in a number of computer languages.⁸⁵ This is an important distinction that must be observed. Having broadly reviewed the standards that are needed, we turn to a specific test for a software patent.

IV. A TEST FOR SOFTWARE PATENT CLAIMS

Previously, four paradigms from the article *Patenting Mathematical Algorithms*⁸⁶ for computer-related inventions were set out: (1) claims to a pure computational method or “algorithm,” (2) claims to programmed apparatus, (3) claims to data structures, and (4) claims to program storage devices.⁸⁷ The first is the most difficult and is the pure “software patent.” Claims to programmed apparatus will be addressed later in the context of the discussion of claims to pure computational methods.

A. *What Is a Pure Computational Method Claim?*

The first trigger for the test should be whether or not the claim is directed (a) to a pure computational method or algorithm method or (b) to an article of manufacture using means plus function clauses and equivalents thereof which should be analyzed as though it actually was directed to a method comprising the functions performed by the claimed means.⁸⁸

Based on the gloss in the case law, a pure computational method or algorithm method means, “Does the claimed invention involve a mathematical algorithm, and if so, is there (a) insignificant post-solution activity and (b) no transformation or determination of physical structure in the claim?” If the answer to the question is “yes,” then the claim is a pure computational method. The test is not intended to permit stretching

85. *Id.* at 1029–30.

86. *See Patenting Mathematical Algorithms*, *supra* note 4.

87. *Patenting Mathematical Algorithms*, *supra* note 4, § 6.03.

88. *Ex Parte Alappat*, 23 U.S.P.Q.2d (BNA) 1340, 1344–45 (Board of Patent Appeals and Interferences 1992); *In re Alappat*, 33 F.3d 1526, 1560 (Fed. Cir. 1994) (en banc); *see also In re Maucorps*, 609 F.2d 481, 486 (C.C.P.A. 1979).

of the concept of structure in order to evade the necessity of satisfying the test.

B. The Test for Patentability of a Pure Computational Method Claim

In order for a software patent claim to a pure computational method or mathematical algorithm to be allowable, the inventor bears the burden of showing:

(a) the pure computational method or mathematical algorithm, which will be collectively referred as a pure computational method, must be an unambiguous specification of a conditional sequence of mathematical steps or mathematical operations for solving a class of problems.⁸⁹ If it is not unambiguous, the claim should be rejected under section 112⁹⁰ as indefinite, and potentially be rejected under section 101.⁹¹

Further, a pure computational method or algorithm method, for purpose of determining whether the specific test in this article is applicable under the patent laws, means: “Does the claimed invention involve a mathematical algorithm, and if so, is there (1) insignificant post-solution activity, and (2) no transformation or determination of physical structure in the claim?”

If the answer to the question is “yes,” then the claim is a pure computational method and must be carefully analyzed in accordance with this test. If not, traditional principles of patent examination and allowability are satisfactory, including principles in the cases requiring substantial post-solution activity or physical transformation. The test is not intended to permit stretching of the concept of structure in order to evade the necessity of satisfying the test in this article.

(b) only a method is being claimed, and any claim for a machine utilizing the method is confined to a specific method and that specific method meets the test below.

(c) the pure computational method is not substantially a natural phenomenon, an abstract idea, a law of nature or the like.

(d) The disclosure must demonstrate that the method utilizes and embeds in software a scientific or mathematical relationship or formula that has already existed, or disclaim a just-discovered relationship, and the disclosure must demonstrate that the method is not the sole method to transform input into the desired output. The claim cannot be so abstract and sweeping as to cover unknown uses of the algorithm (and by

89. Newell, *supra* note 83, at 1024.

90. 35 U.S.C. § 112 (1988).

91. 35 U.S.C. § 101 (1988).

stating its transforming function and so limiting the claim, must claim only known uses) nor may it wholly pre-empt the mathematical formula or algorithm and in practical effect be a patent on the formula or algorithm itself. A disclaimer in the patent to any portion of the method that is substantially a mathematical formula, a natural phenomena, an abstract idea, a law of nature or the like should normally be effective not to pre-empt a mathematical formula or algorithm.

In particular this requirement will exclude fundamental formulae such as $v=\lambda f$. However, the fact that the software method may be more substantially more efficient and cost-effective than a previous method is not indicative that the software is the sole method. If it is substantially a mathematical formula,

(e) the pure computational method must be embodied in a software program.

(f) the pure computational method must achieve a utilitarian function, i.e., a technologically useful application, and in particular the object of that utilitarian function must be patentable under existing law (aside from the existing law on computer algorithms for which this is a new test); in particular, a method of doing business is not patentable. The method must be directed to a technologically useful application. Such an application may be manipulating data for a business purpose. The focus of the claim must be on the manipulation; the “transforming” must be on a data set. The focus cannot be on claiming a business method as the transforming function. A “method of doing business” should be as presently defined under existing case law. Generally, a claim to system for doing business is not patentable, as opposed to a means for carrying out the system,⁹² or as opposed to a method of calculation for the system. An algorithm to find a critical path among an abstract set of input to generate output for a specified business purpose is acceptable because the patent is on the algorithm and the specified business purpose limits the scope of the claim so others can use the algorithm.

(g) the pure computational method must be drafted in terms of the just-referenced utilitarian function to accomplish a stated practical application, which function for a stated practical application is a limitation of the claim. This means that the claim must be substantially in the form of:

92. *In re Patton*, 127 F. 2d 324, 327 (C.C.P.A. 1942) (holding “that a system for transacting business, apart from the means for carrying out the system, is [not patentable subject matter]”).

What is claimed is:

a method embedded in a set of computer-translatable instructions comprising:

[recite steps in terms of present participles: doing something to input, or to input transformed by a prior step]

so that input is [transformed] in conjunction with a computer into [output of a specified character for a specified technologically useful application.]

An alternative form of claim is:

What is claimed is:

a method embedded in a set of computer-translatable instructions comprising:

[recite steps in terms of present participles: doing something to input, or to input transformed by a prior step]

in order to [transform] input in conjunction with a computer into [output of a specified character for a specified technologically useful application.]

Put another way, a patent should be allowable on a method for using a computer to transform “A” to “B” according to a formula or algorithm in order to generate machine readable information to do something of practical value or so that something of practical value is done. The reason for the requirement that the pure computational method be embedded in a set of computer instructions is that if the pure computational method is not utilized on a computer, and if it is a pure computational method, it should not be patentable. This is consistent with existing law.⁹³ Generally, the public has no interest in allowing a series of “hand-run” mathematical steps to be patented.

(h) the input must be a set of numbers of a specific type or character derived from a specific source. The input to the pure computational method must be apparent from the claim, not merely the disclosure, to enable a person reasonably skilled in the art to determine if transformation according the claim of certain input is encompassed in the claim. Merely referring to the input as a set of numbers will not do; it must be a set of numbers derived from a [specified] source of a particular type or character. In some circumstances, it will be obvious from

93. *Mackay Co. v. Radio Corp.*, 306 U.S. 86, 94 (1939), *cited in* *Gottschalk v. Benson*, 409 U.S. 63, 67–68 (1972), *Diamond v. Diehr*, 450 U.S. 175, 188 (1981), and *Arrhythmia Research Technology Inc. v. Corazonix Corp.*, 958 F.2d 1053, 1057 (Fed. Cir. 1992).

the claim what the input must be. For example, if the claim stated: “What is claimed is a method comprising: multiplying a matrix composed of [a set of characteristics of a generally described set of chemicals] [in a particular way that transforms the data set] . . . in order to determine [a proper mix ratio],” the input is sufficiently clear and the designation of particular chemicals or their brand name should not normally be required.

(i) Unless (1) all means of accomplishing a means plus function element are obvious and well-known in the art, (2) the universe of such means plus function(s) is well-known in the art, and (3) the universe is not known to be under development in the art, means plus function clauses or steps specify a well-known function having well-known means to accomplish that function or they should be discouraged or prohibited. A reference to a means for sorting or a means for multiplying matrices is acceptable, i.e., a reference to a matrix multiplication subroutine with an example referenced is proper. A reference to means for simulating artificial intelligence is not acceptable.

Any machine claim must be analyzed in terms of method steps, and the only allowable elements of the machine claim to a computer implemented invention which pertain to an algorithm should be “means of [method step of allowable method claim].” Alternatively, “a machine using the [allowable] method of claim [number]” may be permitted.

As a corollary, any future invention that utilizes a new “means for function” that did not exist at the time of the issuance of the patent on the present invention will fall outside the claim.

Equivalents must be narrowly construed for patents and claims for computational methods. The recent case of *Hilton Davis Chemical Co. v. Warner-Jenkinson Co.*,⁹⁴ interprets the so-called “function-way-result” test established in *Graver Tank & Manufacturing Co. v. Linde Air Products Co.*,⁹⁵ to include “insubstantial differences.” Under this test, the use of a given computational method for a different function will always be non-equivalent; only the use of a substantially similar transforming computational method of a similar set of input for a substantially similar output and function will be an equivalent computational method.

(j) The tests for utility, novelty, and non-obviousness must be met under sections 102 and 103.⁹⁶ Traditional principles are generally useful

94. 62 F.3d 1512, 1518 (Fed. Cir. 1995) (en banc) (holding that independent development is probative of infringement under the doctrine of equivalents), *cert. granted*, 116 S. Ct. 1014 (1996).

95. 339 U.S. 605 (1950).

96. 35 U.S.C. §§ 102–103 (1988).

except to the extent they need to be modified to comport with these standards. Embedding a known method in software is not novel. The preceding discussion on equivalents is useful in examining for novelty as against prior art. Using a known method in a novel way will often not be patentable, though also not be infringing a prior patent disclosing such known method.

(k) There must be a demonstrable actual reduction to practice. The best mode of any means plus function clause must be known in the art or must be submitted on a paper copy or a disk submitted to the PTO, and the best mode of any other aspect of the invention must be submitted on a paper copy or disk submitted to the PTO or a particular module of a software program or algorithm disclosed, cited, and known in the art. This means a demonstrably operable copy. The importance of this test is that “the People” are only willing to award the monopoly if the inventor discloses the invention in a useful way and the disclosure enables the invention to be used for scientific research and experimentation so that the next disclosure may be had.

V. WILL THIS TEST SOLVE THE CONCERNS OF THE ALAPPAT DISSENTERS?

The test set forth above should define the “not patentable” and the “patentable” portion of the spectrum definitively and substantially narrow the gray area in between. The PTO and courts are putting their feet in what could be dangerous waters, and a cautious approach is proper, given the enormous and universal usefulness of some algorithms. Fulfilling the above test will not be easy, and patents will be limited (as they should be) if no substantial post-solution activity or physical transformation is involved.

Before reviewing the *In re Alappat* dissent, the reader is reminded that the majority of the Federal Circuit deciding *In re Alappat* en banc, characterized the invention as relating “generally to a means for creating a smooth waveform display in a digital oscilloscope.”⁹⁷ Based on the statutory language of 35 U.S.C. § 112 ¶ 6, the Federal Circuit held that means plus function elements had sufficient physical structure inherent in an electronic machine to be patentable. The key quotation was:

Although many, or arguably even all, of the means elements recited in claim 15 represent circuitry elements that perform mathematical calculations, which is essentially true of all digital

97. *In re Alappat*, 33 F.3d 1526, 1537 (Fed. Cir. 1994).

electrical circuits, the claimed invention as a whole is directed to a combination of interrelated elements which combine to form a machine for converting discrete waveform data samples into anti-aliased pixel illumination intensity data to be displayed on a display means. This is not a disembodied mathematical concept which may be characterized as an “abstract idea,” but rather a specific machine to produce a useful, concrete, and tangible result.⁹⁸

This author believes that the inherent association of software with a machine will not be a useful analytical tool. While the inventor in *In re Alappat* should receive a patent, it should be because the method is patentable or the machine using a patentable method is patentable. The invention should not be patentable simply because the algorithm/formula/method is embodied in means plus function elements.

*A. The Test for Pure Computational Methods and
the In re Alappat Dissent*

The lengthy dissenting opinion written by Chief Judge Archer in *In re Alappat* (hereinafter the *In re Alappat* dissent) is a useful anvil on which to forge and to test the standard in this article.

The *In re Alappat* dissenters had a series of major complaints about software patents and algorithm patents generally and characterized the Alappat patent as having

arranged known circuit elements to accomplish nothing other than the solving of a particular mathematical equation represented in the mind of the reader of his patent application. Losing sight of the forest for the structure of the trees, the majority today holds that any claim reciting a precise arrangement of structure satisfies 35 U.S.C. § 101.⁹⁹

The first concern expressed was the danger that

[t]hrough the expedient of putting his [a composer’s] music on known structure, can a composer now claim as his invention the structure of a compact disc or player piano roll containing the melody he discovered and obtain a patent therefor? The answer must be no. The composer admittedly has invented or discovered nothing but music. The discovery of music does not

98. *Id.* at 1544 (footnotes omitted).

99. *Id.* at 1552 (Archer, C.J., concurring in part and dissenting in part).

become patentable subject matter simply because there is an arbitrary claim to some structure.¹⁰⁰

Judge Archer rightly complains that “the arbitrary claim to some structure” is not a helpful distinction. I will return to his accurate criticism of the artificial nature of structure below.

When applying the above test, however, Judge Archer’s concern about music being incorporated into structure is belied by the eloquent passage (quoted by him in the *In re Alappat* dissent) of George Curtis, making general observations about patent law in 1873: “The patent law relates to a great and comprehensive class of discoveries and inventions of some new and useful effect or result in matter, not referable to the department of the fine arts”¹⁰¹

The standard set forth in this article adopts the understanding of the courts, patent bar, and Curtis’s suggestion of a century ago that the word “useful” in the patent context means excluding the fine arts, e.g., music on a compact disc is no different than music on a phonograph record. Both a compact disc and phonograph record are patentable—the music is not. Thus, the first concern of the *In re Alappat* dissent is resolved by using the traditional definition of “useful” as excluding the fine arts.

In the course of expressing this concern over music, the *In re Alappat* minority made the following observation: “If Einstein could have obtained a patent for his discovery that the energy of an object at rest equals its mass times the speed of light squared, how would his discovery be meaningfully judged for non-obviousness, the sine qua non of patentable invention? [citation omitted].”¹⁰²

The answer to that question is easy. $E=mc^2$ was non-obvious because newly-pronounced natural laws are almost always by definition non-obvious. The answer is so easy that the non-obviousness does not aid the analysis. Absent constituting pure algebraic manipulation, virtually any new natural law to be encoded in a software patent will be non-obvious.

More important is the second question posed by the *In re Alappat* dissent: “[w]hen is the abstract idea ‘reduced to practice’ as opposed to being ‘conceived?’”¹⁰³ This is fortunately an easy answer in the context of software. An abstract idea is reduced to practice when a program is written or the programming modules or subroutines are designated and the best mode of invention is disclosed. That is the reason for the final

100. *Id.* at 1554.

101. *Id.* at 1551 (quoting G. CURTIS, A TREATISE ON THE LAW OF PATENTS FOR USEFUL INVENTIONS xxiii–xxv (4th ed. 1873)).

102. *Id.* at 1553.

103. *Id.*

requirement in the test above; there must be a depository copy or stated writing in the disclosure of best mode of the software.¹⁰⁴ Absent such a requirement, Judge Archer's critique correctly observes that the public will not know if an actual working reduction to practice has occurred. For algorithms, the idea that the patent application is a constructive reduction to practice is not satisfactory, not any more than it would be for biological material. Without such demonstrable reduction to practice, no patent should be granted because of the algorithmic nature of software.

Returning to the issue of structure, the *In re Alappat* dissent observed, "Applicants sometimes attempt to claim digital-electronic related subject matter by reference to the mathematical function performed by the digital electronic structure."¹⁰⁵ The *In re Alappat* dissent then reached back to *Gottschalk v. Benson*,¹⁰⁶ observing that "the Supreme Court held that claims to a method of converting binary-coded decimal numbers into pure decimal numbers did not recite an invention or discovery within § 101, and thus were ineligible for patent protection."¹⁰⁷ Based on this statement, the *In re Alappat* dissent argued that inventions that are substantially mathematical manipulation should not be patentable.

As stated previously, the author submits that the *Benson* case has to be viewed on its facts. Two of the *Benson* Court's criticisms were that (a) the "mathematical formula that had no substantial practical application except in connection with a digital computer . . . [and (b)] the patent would wholly pre-empt the mathematical formula and in practical effect would be a patent on the algorithm itself."¹⁰⁸ This test evades the first observation by incorporating the concept of transformation of input to technologically useful output to eliminate incorporation of formulae for formulae's sake into software. Second, by imposing a requirement that the method incorporated into the software improve an old method or be a new short-cut, the preemption-of-field issue and "patent-on-formula" issue is eliminated. The narrowness of the author's test addresses Judge Archer's criticism and should eliminate patents being allowed simply because they have a formula embodied on a machine.

104. The issue is less clear if the deposit is handled like a biological material deposit, i.e., stored by the inventor or his designee, stored like a printed copy of the patent at the PTO, or handled like a deposit of biological material. Assuming copies can be obtained at cost, there is an attraction to off-site storage by the inventor who could monitor copies produced, protect the inventor's and/or author's copyright, and control the "fair use."

105. 33 F.3d at 1554 (Archer, C.J., concurring in part and dissenting in part).

106. 409 U.S. 63 (1972).

107. *In re Alappat*, 33 F.3d at 1555 (Archer, C.J., concurring in part and dissenting in part).

108. *Benson*, 409 U.S. at 71–72.

The *In re Alappat* dissenters' and the *Benson* Court's argument that patents should not be allowed because the invention was useful in connection with only one type of device (a computer) is not constructive, and it ignores the line of cases allowing inventions that are useful in connection with only one device. Such a requirement that an invention be useful on more than one type of device would disqualify a whole range of inventions, including phonograph records, which no one seems to seriously argue are unpatentable.

Judge Archer summarizes his analysis of the conclusions of *Benson*, *Flook*, and *Diehr* as follows:

Under [these cases] the posing and solution of a mathematical *Flook* function is non-statutory subject matter. It is non-statutory even if the particular mathematics is limited to performance in digital electronic circuitry or a general purpose digital computer, even if the mathematical operations are alleged generally to have some application in one or various technologies, and even if the solution of the function is said generally to "represent" something of physical or technologic relevance. On the other hand, an invention or discovery of a process or product in which a mathematical operation is practically *applied* may be statutory subject matter. The fact that one element of the claimed process or product is a programmed digital computer or digital electronics performing a mathematical function does not necessarily preclude patent protection for the process or product. In this way, the door remains open to the advancement of technologies by the incorporation of digital electronics. But the mere association of digital electronics or a general purpose digital computer with a newly discovered mathematical operation does not per se bring that mathematical operation within the patent law.¹⁰⁹

The distinction between an invention of a computer containing software and a programmed digital computer being one of several elements in a process seems intellectually unsatisfactory. At the same time, Judge Archer's dissent usefully references that "an invention or discovery of a process or product in which a mathematical operation is practically *applied* may be statutory subject matter."¹¹⁰

109. *In re Alappat*, 33 F.3d at 1557 (Archer, C.J., concurring in part and dissenting in part).

110. *Id.* at 1557 (Archer, C.J., concurring in part and dissenting in part).

Starting from the *Morton* case¹¹¹ and applying the focus of this latter phrase from Judge Archer's opinion, the author's test in this article requires that the method not be a fundamental law, that it be a series of steps that is functional, and that it be limited in its claims to its function (i.e., to transforming data for a stated practical application). Thus, the steps of a particular method claim allowed under this standard, when used for a different non-equivalent function, for a different non-equivalent application, or for a different input set to generate an output for a different purpose, will fall outside the applicant's claim. This leaves the door open for a new invention for another practical application. However, in order to obtain a patent on a new invention outside the existing patent, it will have to be non-obvious.

Next addressed by the *Alappat* dissent is the necessity of determining what was discovered. What was discovered was, according to the majority, "a means for creating a smooth waveform display in a digital oscilloscope."¹¹²

Claim 15, cited by the *Alappat* minority, read as follows:

15. A rasterizer for converting vector list data representing sample magnitudes of an input waveform into anti-aliased pixel illumination intensity data to be displayed on a display means comprising:
 - (a) means for determining the vertical distance between the endpoints of each of the vectors in the data list;
 - (b) means for determining the elevation of a row of pixels that is spanned by the vector;
 - (c) means for normalizing the vertical distance and elevation; and
 - (d) means for outputting illumination intensity data as a predetermined function of the normalized vertical distance and elevation.¹¹³

The initial Board of Patent Appeals and Interferences panel, on appeal from the examiner's rejection, found that "[e]ach clause of the body of [Alappat's] claim 15 recites a mathematical operation and they are recited to operate together to reach a numeric value or pure number

111. *Morton v. New York Eye Infirmary*, 17 F. Cas. 879 (C.C.S.D.N.Y. 1862) (No. 9865).

112. *In re Alappat*, 33 F.3d at 1557-58 (Archer, C.J., concurring in part and dissenting in part).

113. *Id.* at 1558 (Archer, C.J., concurring in part and dissenting in part).

as the end product of the claim.’”¹¹⁴ The reconsideration panel “reasoned that the means for function clauses must be interpreted as covering every structure for performing the recited function, and the burden was on the applicant to prove otherwise.”¹¹⁵ This is consistent with PTO practice and *In re Donaldson Co.*¹¹⁶ Both of these observations appear to the author to be accurate, and within those observations are several keys to the test set forth in this article dealing with both novelty and the restriction on means for function clauses.¹¹⁷ The *Alappat* majority went to some length to use the *Donaldson* grasp of equivalents to hold that the “means for function” clauses included physical structure and made the claim allowable.

The proposition that the means for function clauses inherently include physical structure per se is simply overbroad. The next inventor who has a different formula or algorithm that accomplishes the same function should not fall within the means for function clause. That is not the public’s intent and it overgrants the monopoly in the area of pure computational methods. As the reconsideration board in *Alappat* did correctly observe, “the claim was to every structure for performing the recited mathematical functions, and the claim was to be analyzed as though it actually was directed to a method comprising the functions performed by the claimed means.”¹¹⁸

The test in this article desires to prevent the pure computational method being disguised as a machine claim and to prevent the machine claim from being analyzed differently than a method claim would be analyzed.¹¹⁹ In part, this is to assist in classification and analysis of prior art.

Thus, the test in this article would analyze subparts (a)–(d) of *Alappat* claim 15 by rewriting them as a method claim and then applying the test:

114. *Id.* at 1559 (Archer, C.J., concurring in part and dissenting in part).

115. *Id.* at 1560 (Archer, C.J., concurring in part and dissenting in part). *See also Ex parte Alappat*, 23 U.S.P.Q.2d (BNA) 1340, 1342 (BPAI 1992).

116. 16 F.3d 1189, 1193–95 (Fed. Cir. 1994) (holding that the Patent & Trademark Office must apply the means-for-function test to all such claims language); *see* MPEP, *supra* note 11, §§ 2106.02, 2181, 2186.

117. *See supra* note 97 and accompanying text; *In re Alappat*, 33 F.3d at 1540–41 (ruling that a computer operating on software is patentable).

118. *In re Alappat*, 33 F.3d at 1560.

119. *See id.* at 1568.

ALAPPAT CLAIM

- (a) means for determining the vertical distance between the endpoints of each of the vectors in the data list;
- (b) means for determining the elevation of a row of pixels that is spanned by the vector;
- (c) means for normalizing the vertical distance and elevation; and means for outputting illumination intensity data as a predetermined function of the normalized vertical distance and elevation.
- (d) means for outputting illumination intensity data as a predetermined function of the normalized vertical distance and elevation.

PROPER METHOD CLAIM

- (a) determining the vertical distance between the endpoints of each of the vectors in the data list;
- (b) determining the elevation of a row of pixels that is spanned by the vector;
- (c) normalizing the vertical distance and elevation; and
- (d) outputting illumination intensity data as a predetermined function of the normalized vertical distance and elevation so that . . . [to be discussed momentarily]

The claim is and should be first analyzed as a method claim only. The inventor should have been entitled to a patent for what he invented, which was not an oscilloscope or any device, but a good and useful computational method using certain input that had a specific result in connection with an oscilloscope.

The other concern that the standard in this article deals with concerns the preamble. The *Alappat* majority holds that the word “rasterizer” in the preamble is not a mere “field-of-use limitation” but limits claimed subject matter to the production of “output illumination data.”¹²⁰ Preambles of claims are not always interpreted to be a narrow limitation on the claim. What the test in this article proposes is not a mere “field-of-use” limitation; the test in this article requires that the

120. *Id.* at 1544.

limitation the majority inferred in the *Alappat* preamble be a limitation actually stated in the claim. The contrast is as follows:

ALAPPAT CLAIM

A rasterizer for converting vector list data representing sample magnitudes of an input waveform into anti-aliased pixel illumination intensity data to be displayed on a display means comprising:

- (a) means for determining the vertical distance between the endpoints of each of the vectors in the data list;
- (b) means for determining the elevation of a row of pixels that is spanned by the vector;
- (c) means for normalizing the vertical distance and elevation; and
- (d) means for outputting illumination intensity data as a predetermined function of the normalized vertical distance and elevation.

PROPER METHOD CLAIM UNDER PROPOSED TEST

A method for creating a smooth waveform display in a digital oscilloscope comprising:

- (a) determining the vertical distance between the endpoints of each of the vectors in the data list;
- (b) determining the elevation of a row of pixels that is spanned by the vector;
- (c) normalizing the vertical distance and elevation; and
- (d) outputting illumination intensity data as a predetermined function of the normalized vertical distance and elevation in order to convert vector list data representing sample magnitudes of an input waveform into anti-aliased pixel illumination intensity data to be displayed in smooth waveform on a display means in an oscilloscope. [alternatively: so that vector list data representing sample magnitudes of an input waveform are converted into anti-aliased pixel illumination intensity data to be displayed in smooth waveform on a display means in an oscilloscope.]

This form and claim makes the method patentable, so the inventor obtains the monopoly desired. The patent claims are limited to and by their functionality in transforming certain input. At the same time, any person who desires to use (a) a first ALU [arithmetic logic unit]; (b) a second ALU; (c) two barrel shifters; and (d) a ROM [read-only-memory] and use the same calculation methodology in another function for a different application will be free to do so. The public will have the benefit of the disclosure, but the hurdle to future use, and the hurdle of inventing around the *Alappat* patent will not be raised too high, which would be against the public interest.

The *Alappat* patent, as it was allowed, is attractive in a policy sense because its disclosure allows others to study it for scientific, experimentation, and research purposes and to design around it. This comports with the goals of the patent system, even if the patent bar is tempered in its enthusiasm for those goals. Though not referenced by the dissent, surely the fact that there was a detailed disclosure must have comforted the majority.

The dissent also criticizes the majority observation that “if the claimed ‘rasterizer’ were equivalent to a ‘general purpose digital computer’ programmed to perform the calculations performed by the rasterizer, such programmed computer would be the invention of a ‘new machine’ within § 101.”¹²¹

This latter statement illustrates that the *Alappat* majority’s “structure analysis” tends to be unproductive. Reprogramming a computer in a useful way is no person’s common understanding of a new machine. Machine claims do not make much sense where pure computational methods are involved, and there is no other physical structure outside the computational machine using the algorithm. The rule in *In re Maucorps*¹²² that claimed apparatus was non-statutory even though it referred to a disclosed dedicated hard-wired circuit appears correct in the software patent context;¹²³ a hard wired system that is non-programmable can just as well capture a formula and such a system should not be patentable, however narrow the patent may seem to be. Judge Archer’s observation about a newly programmed player piano not being a new machine is a particularly appropriate analogy to why an algorithm is only a series of steps, only a method.

121. *Id.* at 1561.

122. 609 F.2d 481 (C.C.P.A. 1979).

123. *Id.* at 485.

B. A Digression to a Criticism Not Raised in In re Alappat: Separating Out Business Methods from Algorithm Patents

One criticism that the *Alappat* majority did not raise to claims of a pure computational method was the “business method” problem. Methods of doing business should not be patentable, whether obvious or disguised in the claims. In the context of software, this is not a distinction that is any easier to make than in a non-software context. However, as an example, a new software program for predicting the weather that will use a set of inputs and transform and manipulate those inputs into output is very technologically useful. Yet, there is no physical transformation involved. We still cannot change the weather. Contrast this weather prediction method with the method of optimizing bids addressed in *In re Schrader*.¹²⁴ Taken as a whole, the latter method seems like a method of doing business, which is and should continue to be non-patentable. Judge Newman’s characterization in the *Schrader* dissent of a “technologically useful result” is a helpful test to separate out business methods from patentable computational methods.¹²⁵ In *Schrader*, Judge Newman concludes that “historical distinctions between a method of ‘doing’ business and the means of carrying it out blur in the complexity of modern business systems.”¹²⁶ This seems an overstatement. Judge Newman, while criticizing the business method exception to patentability, stated that “a system for transacting business, separate from the means for carrying out the system, is not patentable subject matter”¹²⁷

Rearranging data or performing an analysis should be a requirement to a patent. Demanding that the result of that effort—the practical, technologically useful application—be stated as a limit to the claimed algorithm should render the fuzzy concept of doing business on the spectrum of patentability to a narrow range. For example, as in *Schrader*, transforming data to guide a robot, or to produce a data table useful in a technical application, seems much different than claiming a function producing an optimal bid scheme. The difference is a focus in the claim on the manipulation of specified input (in the “transforming” language of the claim) for a specified purpose. Thus, by requiring that the claim form be written using present participles of “action verbs,” or

124. See, e.g., *In re Schrader*, 22 F.3d 290 (Fed. Cir. 1994) (rejecting the inventor’s patent application for a method of competitively bidding on a group of related items such as contiguous tracts of land).

125. *Id.* at 297 (Newman, J., dissenting).

126. *Id.* at 298 (Newman, J., dissenting) (citation omitted).

127. *Id.* at 298 (Newman, J., dissenting); see also *In re Patton*, 127 F.2d 324, 327 (C.C.P.A. 1942).

of transitive verbs in present participle form, what data are manipulated becomes very important. Most modern thinking would regard computer science, despite its major applications to business, as a technological field. Patent law should be able to maintain the focus on transformation. In *In re Schrader*, the claim was made to “a method for competitively bidding on a plurality of related items.”¹²⁸ This claim was a business method by its very wording.¹²⁹ Further, the inventor in *In re Schrader* did not specify steps of transformation of data. One step was “offering said plurality of items to a plurality of potential bidders.”¹³⁰ The transforming language needs to be focused on manipulation of data of specified input. The word “offering” is not a transforming verb or a verb of movement. Contrast this with the word “shifting,” implying a movement of numbers or data. Note also that “shifting” can be a non-mathematical step.

Under the test in this article, Schrader could have obtained a patent for a method of selecting an optimal number, comprising the steps of (1) entering a series of bids into a data record, (2) indexing each of said bids, and (3) applying a specified algorithm to maximize the sum of bids selected from said data record in order to select the highest combination of bids maximizing revenue to the seller of property.

What Schrader wanted to patent was the idea of the overall system of “[a] completion identifying a bid for all of said items at a prevailing total price” without giving the details.¹³¹ Schrader did not specify the algorithm for which the public should be willing to grant a patent and rightly was denied a patent.

C. A Return to *In re Alappat*'s Criticisms—Examination Difficulties

Another criticism that *Alappat* did not raise, which the Supreme Court cited in *Gottschalk v. Benson*,¹³² was the inability to examine software patents because of a lack of a classification technique and the proliferation of prior art. This seems unpersuasive. If claims are confined to transformation directed to a function, classification can be dealt with in terms of function. While there will be many new functions, the proliferation of classes and subclasses occurs in most art fields. The proliferation of prior art will restrict inventions to the required non-

128. *Schrader*, 22 F.3d at 291 (rejecting the inventor's patent application for a method of competitively bidding on a group of related items such as contiguous tracts of land).

129. *Id.* at 291–92.

130. *Id.* at 292.

131. *Id.* at 291–92 (rejecting the inventor's patent application for a method of competitively bidding on a group of related items such as contiguous tracts of land).

132. 409 U.S. 63, 72 (1972)

obvious improvements. Art will have to be that which is known to the reasonably skilled practitioner in the art, which will lead to no more or less factually intensive inquiries than presently exist.¹³³

D. The Test to Patents in Recent Cases

If the test set out in this article were applied to some of the patents in the leading cases, many of them would fail. Flook's invention, which incorporated data from a known method of processing hydrocarbons into a computer triggered alarm system, should have failed for lack of novelty or for obviousness. Assuming the method of transforming data was novel, Diehr's patent to use a computer to translate inputs into a computer-implemented trigger for an injection machine in a rubber curing process should be patentable based on the fact that data was transformed to open a machine and produce rubber. Maucorps' patent should fail as a business method. The patent claimed in *Gottschalk v. Benson*¹³⁴ also should probably have failed. It is not clear that there was any other way of doing what was claimed; i.e., it was a basic formula, and there was no functional limitation. Furthermore, the formula did not appear to have been non-obvious over existing hand-calculated formulae. The claims in *In re Grams*¹³⁵ to "[a] method of diagnosing an abnormal condition"¹³⁶ that is "applicable to any complex system"¹³⁷ should have failed for insufficient limitation to a particular function.¹³⁸

Similarly, Trovato's claims, which were meant to cover a method for solving the "critical path problem," should be rejected because they claimed an algorithm without narrowing their patent to a technologically useful application.¹³⁹ In doing so, the algorithm would then be available for the public to use for other transforming functions to technologically useful ends. Trovato's claims are a good illustration of semantics gone awry as inventors attempt to disguise their overbroad algorithm. For example, the first claim in Trovato's application refers to

133. For example, in the field of electrogalvanic cells, when the Leclanché cell was invented (with similar chemistry to your basic present-day flashlight battery), one could hardly have predicted the proliferation of types of electrogalvanic cells, foreign and domestic, patented and unpatented, that would exist within 100 years, much less the new crop in the last twenty years.

134. 409 U.S. 63 (1972)

135. 888 F.2d 835 (Fed. Cir. 1989).

136. *Id.* at 836.

137. *Id.* at 840.

138. See *Patenting Mathematical Algorithms*, *supra* note 4, § 6.03.

139. *In re Trovato*, 42 F.3d 1376, 1377 (Fed. Cir. 1994), *vacated*, 60 F.3d 807 (1995).

a data structure and describes it as “[a] method for determining motion of an object”¹⁴⁰

Those in favor of patenting software may criticize the test in this article as too limiting on the scope of protection, but that is the purpose of the test. Patenting algorithms presents a high risk of granting an overbroad monopoly. A patent on a fuzzy logic, expert system, or artificial intelligence program to play chess (i.e., translating an input of a chess game at a certain move into an optimal output of the next move) seems eminently desirable because it would leave the rest of the world free to enjoy the benefits of the software and to learn valuable lessons from it. The inventor, however, would have to disclaim a series of basic formulae for eliminating certain fields of non-optimal chess moves. This may eviscerate portions of what the inventor would like to see protected, but this demonstrates the point of the test. If a proposed software patent is too basic, it should not be patentable for the policy reasons articulated above.

VI. CLAIMS TO PROGRAMMED APPARATUS

Initially, the author felt only method claims to pure computational methods should be allowed. However, in light of the majority’s opinion in *Alappat* and in consideration of the policy goal of gaining disclosure through a patent monopoly, the author believes a limited machine claim is allowable without overgranting a patent monopoly.¹⁴¹

The form should be similar to that which was not allowed in the initial appeal in *In re Beauregard*.¹⁴² “An article of manufacture comprising a general purpose computer using the method [executing the method] of Claim X.”¹⁴³ Alternatively, a machine with elements expressed as “means of [allowable method step of an allowable method claim] would be patentable.”¹⁴⁴

140. *Id.* at 1377. Perverse as it may seem, had Trovato then followed the format of this article, Trovato may have obtained a patent, but not for the field of art which she desired, thus giving the public the disclosure without any practical diminution of the public domain.

141. Not with any enthusiasm, however, because software is always a series of steps; any other characterization or analysis of the invention and claim supposes a false predicate which will surely end in incomprehensible knots of holdings in cases. *See generally* Newell, *supra* note 83. Patenting the method as a machine can only be for the suspicious purpose in this context of overextending the patent monopoly beyond what should be patentable—the mere method.

142. *In re Beauregard*, 53 F.3d 1583 (Fed. Cir. 1995).

143. *See Patenting Mathematical Algorithms*, *supra* note 4, § 6.03(d).

144. *Id.*

The purpose of the suggested form is to focus the patent and its equivalents on what should be focused on: the public is prepared to grant a focused patent right using an algorithm for a specific purpose in a computer if it is a patentable method and if the means plus function steps correspond to a patentable method.

First, the courts should be particularly careful to analyze an allowable method claim and then determine if the machine-using-method claim follows the steps of the allowable method claim. Any expansion of the pure computational method claim should not be allowed in the context of a machine claim, and, in particular, method claims disguised as machine claims should be considered especially dangerous. Second, in order not to stifle large areas of technology by accidentally granting a patent monopoly over a formula, rarely should an equivalent of a machine-by-method invention be inferred (just as equivalents of the pure computational method claim should be only recognized on a limited basis, as discussed earlier).¹⁴⁵ With respect to any new means plus function element, any new invention that accomplishes the means plus function in a novel way should automatically fall outside the claim of the prior pure computational method patent. This concept is necessary to prevent a means plus function element of a pure computational method from entirely preempting a different pure computational method invention with a new and different way of accomplishing the means plus function. With these careful strictures, a pure computational method machine claim should be sufficiently narrow to protect the public while still encouraging disclosure.

VII. CLAIMS TO DATA STRUCTURES

The placement or storage of data in a data structure should be governed by the standard discussed above for pure computational methods embedded in software programs. The actual data should be governed by the copyright laws. This discussion is focused on claims to methods that are pure algorithms for the arrangement of data.

A data structure is an implementation of an organizational structure imposed on a set of data capable of being stored in an electronic medium. A straightforward example of a data structure is a typical university student information processing record structure. Each student will have information in a field in this nested data structure. Its utilitarian characteristic is that it allows us to generically refer to subsets of records. An example of a data structure follows:

145. See *supra* note 98 and accompanying text.

CAMPUS
COLLEGE
NAME
ADDRESS
ZIP
TEAM
CLUB
COURSE OF STUDY
COURSE (1)
GRADE
PROFESSOR
COURSE (2)
GRADE
PROFESSOR
COURSE_AVAIL
PROFESSOR
NO_STUDENTS

For instance, if I choose to sort the variable “PROFESSOR” with a precursor designating which “PROFESSOR” field I am addressing, the contents of each record in the field called “PROFESSOR” will be sorted. Alternatively, I could sort only the “PROFESSOR’s for “COURSE (1).” I can also search for a particular name anywhere in the data structure as one would often desire, without referring to the generic label. Alternatively, I may search in a particular field of the data structure for a particular name and can specify one or more levels in which to search.

The largest use of computers is not for word processing nor for mathematical processing (in the sense of applying a formula to a set of numbers and performing a calculation for some purpose). The largest use of computers is for *storing and processing information*. Storing involves placing data in a location where it can be retrieved and processing involves manipulating a set of data according to a formula. The above data structure at first looks like a matrix, but, in fact, it is a complex, multi-dimensional linked matrix set. In the proposed PTO Guidelines and in the Manual of Patent Examining Procedure § 2106, it is proposed to classify as non-statutory any novel arrangement that is “a compilation or arrangement of data, independent of any physical element.”¹⁴⁶ This formulation is too narrow and extinguishes a huge area of information technology dealing with how to efficiently store informa-

146. *Proposed Guidelines*, *supra* note 10, at 28,779; MPEP, *supra* note 11, § 2106. See also *Amended Guidelines*, *supra* note 12, at IV.B.1.b.

tion in computer memory and, after processing it, how to store it efficiently for access by laypersons. Neither the contents of the data nor the business method of how the data is arranged in a particular order should be patentable. However, the method of how data gets stored in a particular order so that it can be manipulated more easily should be patentable, provided the method of storage meets the test described in this article. In addition, if the method of storage is claimed and an algorithm of transformation is claimed given that method and array of storage, then the algorithm of transformation should be patentable if it meets the standard for pure computational methods.

The author submits that the recent *In re Lowry*¹⁴⁷ holding (allowing a patent to a data structure) is on the right track and seems to meet the standard of this article. The *In re Warmerdam*¹⁴⁸ invention of a bubble hierarchy for collision avoidance appears to meet the test in this article if the application and claim is limited to what the opinion stated was “controlling the motion of . . . robotic machines.”¹⁴⁹ Similarly, in *In re Trovato*, if the claim is for a novel method of storage directed to permitting a critical path to be found among the data and is limited to a particular use of the data and a particular type of input, a patent should be allowable.¹⁵⁰ By contrast, *In re Schrader* presents a claim to an algorithm but recites no technologically useful application, merely a system or method of bidding.¹⁵¹ Had Schrader presented a claim including his method for transforming a set of inputs to a certain output for a specified purpose, he might well deserve a patent. However, the protection and right of exclusion he sought was much broader and rightly disallowed.

A method of arrangement does not fail on the problem of pure formula. Arranging inherently requires a formula to direct where data will go. However, the arranging function, unlike a formula, is not to transform the data or input itself but to leave a footprint to be electronically reread. Thus, method claims should be allowable as non-formulaic (assuming compliance with the test in this article) if the claim is limited to the objective of arranging data input for rereading according to the following claim model:

147. 32 F.3d 1579 (Fed. Cir. 1994) (holding that claims were not analogous to printed matter, were not obvious, and were not anticipated by prior art).

148. *In re Warmerdam*, 33 F.3d 1354 (Fed. Cir. 1994) (finding five claims void of statutory subject matter and one claim sufficient to pass the definiteness test).

149. *Id.* at 1355.

150. *In re Trovato*, 42 F.3d 1376 (Fed. Cir. 1994).

151. *In re Schrader*, 22 F.3d 290 (Fed. Cir. 1994).

A method for arranging data in a memory storage device, the method comprising:

- arranging data in said memory storage device according to step 1;
 - arranging data in said memory storage device according to step 2 [perhaps by a certain formula];
 - and finally arranging data in said memory storage device according to step 3,
- so that [recite particular utility or function accomplished by such arrangement in the memory storage device].

The first clause restricts the claim's scope to a memory storage device, as does the requirement that the arranging of data occur in a memory storage device. The "so that" clause is intended to recite the function accomplished and to narrow the claim. The arranging could occur in conjunction with the device so long as the data ends up in the device and so long as the "so that" clause narrows the scope of the claim to the function accomplished in the memory storage device.

VIII. CLAIMS TO PROGRAM STORAGE DEVICES

Program storage devices are analogous to data structures, but they may raise special problems because program storage device patents usually are claimed in connection with physical structure. One might assume that the physical structure in conjunction with a method should be patentable, yet the form of a claim to a program storage device is an easy disguise for a claim to a pure formula for acting on data.

In our world of discussing how big our hard disks should be and how much RAM we need, program storage devices (including hard internal disks, floppy transportable diskettes, and compact discs) are in our everyday vocabulary. Conceptually, we should want to allow inventions that store computer readable code or data in novel ways or use a novel design on a physical structure. New inventions in the area of program storage devices are much like a new form of phonograph records. Although they have at least some physical structure involved, the magic or the novelty to a layman's eye is that little ones and zeroes can be placed in them.

Program storage devices are clearly utilitarian. Thus, the proposed standard of "utility only" for patentability in *Patenting Mathematical*

*Algorithms*¹⁵² is not a useful standard for the area of program storage devices because it is too broad.

Program storage devices are frequently referenced in a step in a method claim, temporarily committing an input to a program storage device in preparation for a next step that will act on the input so stored. Because the public does (and should) desire disclosure of novel, useful, and non-obvious program storage devices, the test of patentability should focus on novelty and non-obviousness, using traditional patentability analysis. Due to the close association with mathematical algorithms, the only qualification is that if a claim recites a means for function, any claims for the device must recite a means for function which is in a relatively well-understood and well-developed area of computer technology.

The content of the program storage device in terms of method steps for transforming input into output must be subject to whatever test for pure computational methods the courts and the PTO ultimately adopt. The author of *Patenting Mathematical Algorithms* characterized claims 55 and 66 of *In re Beauregard*¹⁵³ as follows:

55. A program storage device readable by a machine and encoding a program of instructions for executing the method steps of a specified one of claims 1 through 4.

66. An article of manufacture comprising a computer usable medium having computer readable program code means embodied therein for causing a [general description of function], the computer readable program code means in said article of manufacture comprising:

(a) computer readable program code means for causing a computer to effect [description of first specific function];

(b) computer readable program code means for causing the computer to [description of second specific function ‘tied’ to first function] and

(c) computer readable program code means for causing the computer to [description of third specific function ‘tied to first and/or second function’].¹⁵⁴

152. See *Patenting Mathematical Algorithms*, *supra* note 4, § 6.04.

153. 53 F.3d 1583 (Fed. Cir. 1995) (holding that the printed matter doctrine did not apply to computer programs.)

154. See *Patenting Mathematical Algorithms*, *supra* note 4, § 6.03 (citations omitted).

Accepting the author's characterization of the claims for discussion purposes, claim 55 is not problematic if claims 1 through 4 meet the test set forth in this article. Claim 55 is a classic machine-implementing method claim. The question is whether or not the method is allowable, and the test for a pure computational method should be applied.

Claim 66 is considerably more dangerous. If elements (a), (b), and (c) in claim 66 contain algorithms for transforming data instead of algorithms that are restricted to the function of placing data in a program storage device, the unrestricted function component of these elements carves out too broad a monopoly. Applying this article's test, the claim should fail unless: (a) each function is well-known itself; (b) the claim is limited to its functional purpose (in order to store computer readable program code in a particular way); and (c) overall, the objective of the combination of method steps ("the particular way") implicit in the article of manufacture claim is novel, utilitarian, and non-obvious and meets the test in this article. If those tests are not applied, a patent could be allowed for storing a particular formula or formulae in a program storage device. A patent to a program storage device is appropriate for how data is stored in the electronic or physical structure but not for the storage itself or the use of the formula. To draw an analogy to music on a compact disc, the way the music is embedded on a compact disc is patentable; the music itself is not. Also, music per se does not fit within the phrase "description of function."¹⁵⁵

Accordingly, this author believes the Board of Patent Appeals and Interferences should not find the claim in *Beauregard* per se disallowed but should render a decision giving guidance, remand the claim(s), and focus on restricting the claim to what it is and should be: a program storage device with the necessary and associated algorithm to store data in a particular way. If the method of storage contemplated in *Warmerdam* is directed to a particular method in connection with a specific object of the function of such storage, then the claim of a machine having a memory with data in a hierarchy generated by this method should be allowable.

If a formula (which is how we foundered on the danger of software patents in the first place) cannot be claimed per se but can only be claimed in conjunction with either a device or a method of storing data, then the formula and the data itself remain available to the public. Recall that the essence of every software program is a series of mathematical formulae directing a complex circuit to perform in certain ways. If, as the proposed test above requires, those steps must be di-

155. *Patenting Mathematical Algorithms*, supra note 4, § 6.03.

rected to a function and are further tied to a memory structure, and no claim is made to the pure formula, then traditional principles of patent analysis should suffice to protect the public.

IX. CRITIQUE OF THE PATENT OFFICE MPEP GUIDELINES AND THE PRIOR PROPOSED GUIDELINES FOR COMPUTER-IMPLEMENTED INVENTIONS

A. General Observations

On Friday, June 2, 1995, the PTO published a “Request for Comments on Proposed Examination Guidelines for Computer-Implemented Inventions” and “Guidelines for Examination of Computer Implemented Inventions.”¹⁵⁶ In the latest revision of the Manual of Patent Examining Procedure published in September 1995, the subjects of computer-implemented inventions and the necessary disclosure are discussed.¹⁵⁷

The expressed intent of the proposed Guidelines was:

to assist [Patent] Office personnel in their review of applications drawn to computer-implemented inventions . . . [and to] respond to recent changes in the law that governs the patentability of computer-implemented inventions, and [to] set forth the official policy of the Office regarding inventions in this field of technology.¹⁵⁸

The MPEP and the Guidelines appear to restrict software to process patents only, although this intent is unclear.¹⁵⁹ The most important part of the publications may be the “Notes on the Guidelines” and the comments in the MPEP.¹⁶⁰ If the software is claimed without any connection to, or statement of, the context of a process or machine, the Guidelines restrict software to the realm of copyright.¹⁶¹ Such a claim is classified as a non-statutory process. “The specific word or symbols that constitute a

156. *Proposed Guidelines*, *supra* note 10, at 28,778.

157. MPEP, *supra* note 11, §§ 2106–2106.02. The PTO recently made additional modifications to its examining procedures for computer-related inventions, but these have not yet been published in the Federal Register. *See Amended Guidelines*, *supra* note 12.

158. *Proposed Guidelines*, *supra* note 10, at 28,779.

159. *See* MPEP, *supra* note 11, § 2100–5; *Proposed Guidelines*, *supra* note 10, at 28,779–81 (referencing computer program-related elements as specific structure corresponding to an element defined in terms of means plus function).

160. *Proposed Guidelines*, *supra* note 10, at 28,780; MPEP, *supra* note 11, §§ 2106–2106.02; *see also Amended Guidelines*, *supra* note 12, at IV.B.1.

161. *See Proposed Guidelines*, *supra* note 10, at 28,780 (stating that words in a program are expression but still should be rejected under section 103 for obviousness); MPEP, *supra* note 11, § 2106; *see also Amended Guidelines*, *supra* note 12, at IV.B.1.

computer program represent the expression of the computer program and as such are a literary creation A claim in this format should also be rejected under Section 103, as being obvious over the known machine-readable storage medium standing alone.”¹⁶²

Pure data structures which do not direct a computer’s operation and “creative or artistic expressions” (e.g., a work of music, art, or literature) encoded on a “known machine-readable storage medium” are non-statutory.¹⁶³ The restriction that the structure be encoded on a known medium appears unnecessary and is a source of mischief.

Formulae, per se, are a non-statutory claim under the Guidelines: “A claim to a method consisting solely of the steps necessary to converting one set of numbers to another set of numbers without reciting any computer implemented steps would be a non-statutory claim under this definition.”¹⁶⁴ The MPEP contains similar language which classifies a process that does nothing more than manipulate abstract ideas or concepts as non-statutory.¹⁶⁵

Embedding the software in hardware, according to the PTO, does not necessarily resolve the issue of patentability in the inventor’s favor. In the Guidelines, the next sentence after the previous cited passage reads: “This [the method of converting numbers] includes the software and any associated computer hardware that is necessary to perform the functions directed by the software.”¹⁶⁶ In the MPEP, we are left with a negative pregnant: “what if the claim to a method consisting solely of the steps necessary to converting one set of numbers to another”¹⁶⁷ does recite a computer implemented step?

The Guidelines do provide that “a claim that is cast as a computer program but which then recites specific steps to be implemented on or using a computer should be classified as a process.”¹⁶⁸ In that statement, the Patent Office seemed to imply that it was ready to allow a set of steps to be patentable. The MPEP Guidelines, however, do not support such a conclusion. Pure computational methods do not appear to be allowable as is reflected in the prior paragraph.

162. *Proposed Guidelines*, *supra* note 10, at 28,780. *See also* MPEP, *supra* note 11, § 2106; *Amended Guidelines*, *supra* note 12, at IV.B.1.a.

163. *Proposed Guidelines*, *supra* note 10, at 28,779; *see also* MPEP, *supra* note 11, § 2106; *Amended Guidelines*, *supra* note 11, at IV.B.1.a.

164. *Proposed Guidelines*, *supra* note 10, at 28,780; *see also* *Amended Guidelines*, *supra* note 12, at IV.B.1.a.

165. MPEP, *supra* note 11, § 2106. Note the unproductive use of the shibboleth “abstract ideas or concepts” used to disqualify the computer process.

166. *Proposed Guidelines*, *supra* note 10, at 28,779.

167. MPEP, *supra* note 11, § 2106.

168. *Proposed Guidelines*, *supra* note 10, at 28,779.

The required disclosures concentrate on using functional block diagrams. They focus on the need to determine a block without undue experimentation. The MPEP contrasts one case which “found that the amount of experimentation involved was reasonable where a skilled programmer was able to write a general computer program, implementing an embodiment form, within 4 hours” with another case which held that from one and a half to two man years of program development was “clearly unreasonable.”¹⁶⁹ Overall, however, the disclosure requirement is somewhat amorphous and will likely spawn litigation over the sufficiency of disclosure. This is because there is no requirement that a program actually be created or that the blocks be prior art or obvious of creation. “Bugs” are a major problem in software and computer-implemented inventions, and working them out of an otherwise obvious system or block so that the software is basically functional can be a major project. The danger is that once a patent is granted in the novel area of software technology, regardless of whether the software invention works, the patent is a bludgeon to competitors. Given the novelty and breadth of algorithms, more care in preventing premature patents is appropriate. This is not meant to exclude inventions featuring relatively minor problems. An easy example of a recent bug was the incompatibility of certain applications which work with the Windows 3.1™ operating system but do not function with the Windows 95™ operating system.¹⁷⁰ Obviously the system worked for patent purposes, whatever its faults for some users. By contrast, the Pentium™¹⁷¹ chip problem with floating point calculations in 1995, which actually did not function for certain mathematical processes, was arguably dysfunctional, although query whether any examiner could have discerned the difficulty.

Disclosure is even more important because means plus function clauses tend to be broad and are freely usable. The PTO and the *Alappat* majority were sidetracked onto the means plus function analysis because of the Supreme Court’s reference to structure. The breadth of means plus function clauses and the intellectual deception of not using method analysis of computer-implemented inventions will likely cause some inappropriate fluctuation in analysis to reconcile the overbroad means plus function clauses with a difficult analysis tool. The means plus function is one of the best ways to not only obtain claim breadth

169. MPEP, *supra* note 11, § 2106.02 (citing *Hirschfeld v. Banner*, 462 F. Supp. 135 (D.D.C. 1978) and *White Consol. Indus. v. Vega Servo-Control*, 214 U.S.P.Q. (BNA) 796 (S.D. Mich. 1982), *aff’d on other grounds*, 713 F.2d 788 (Fed. Cir. 1983)).

170. Windows and Windows 95 are trademarks of Microsoft, Inc.

171. Pentium is a trademark of Intel Corp.

but also, in a computer-implemented invention context, to make the least disclosure.

B. Specific Comments and Proposed Revisions to the Guidelines

The author's view is that the MPEP and guidelines appear to resolve the *Beauregard* issue and allow patents on program storage devices, although data structures are not addressed specifically. Programmed apparatus, which looks like a machine, is clearly allowed, although the MPEP provides that implementing a non-statutory process on a machine does not create an allowable invention. However, the MPEP does not deal with achieving the public's objective of "disclosure in exchange for a brief monopoly"¹⁷² to allow the patentability of new programs to manipulate data, nor does it deal with the line between computational methods and computational methods programmed on a machine, i.e., programmed apparatus.

In the MPEP and the Proposed Guidelines, the Patent Office apparently fails to recognize that there are innovative ways to manipulate numbers for certain purposes. There is no statutory reason nor is there any policy reason not to allow patents of software for specific purposes. Moreover, in some instances, the copyright laws permit mask works, and no public disclosure is achieved. Even if a work is not filed as a mask work but is filed in observable source code (i.e., machine code) in the depository copy, it is the steps of thinking embedded in the source code which the public would like to know. Program documentation—the explanation of how the program is assembled—allows us to know how to use the program and allows us to improve it. That revelation will only be best achieved through the patent laws.

The earlier draft of this article submitted to the PTO as a comment contained an appendix with suggested changes to the Proposed Guidelines. Because the Proposed Guidelines are now final in the MPEP, the litmus test for software patentability will be in the courts. The concepts in this article are difficult to fit into the present MPEP format. Realistically, the Guidelines should be recast and formatted somewhat differently. As Professor Newell notes, the present model is slightly broken for purpose of software patents,¹⁷³ and the PTO needs to rethink how it organizes the format of the MPEP and the Guidelines for computer-implemented inventions.

172. *Graham v. John Deere Co.*, 383 U.S. 1, 9 n.2, 10 n.3 (1966); see also *id.* at nn. 9–10 (discussing Thomas Jefferson's view of patents as a reward of limited protection in exchange for sharing information).

173. See Newell, *supra* note 83, at 1034 (concluding that present models for evaluating patentability of algorithms are insufficient).

X. CONCLUSION

A. A Vignette in Support of Caution

This article has suggested that the Patent and Trademark Office and the courts should allow patents on pure computational methods in software if appropriate restrictions are embedded by the inventor in the claims. The public wants to know the innovative means of accomplishing certain objects by certain computer programs so that others will use that knowledge to find different and better means to build on those programs for the betterment of society.

This is a controversial subject, to say the least, and in advocating that a new course be set, it is appropriate that the following observation be shared. Otherwise, readers may be persuaded the issue can, so to speak, run before the wind with the spinnaker out.

Without the intent of adding fuel to the fire by impairing DNA patentability, the contrast of the current patentability of DNA sequences and the lack of patentability of software programs illustrate why this area is so difficult and why a balance must be struck.

Recent research suggests that DNA has an inherent capability to reprogram itself. The research on artificial intelligence and the function of sleep suggests that sleep and dreaming during sleep are functions in which our minds (complex computers) generate random sets of images. When we wake up, our subconscious views the world around it and attempts to reconcile the random sets of images with the reality of the world. It may only partially succeed, and, in our next sleep, our subconscious thinks about more random sets of images. However, it is more closely focused on what we just saw, having discarded the completely non-conforming images of the night before from consideration.

Computers are now being set to the same function. An object is shown to the computer which digitizes it. The computer generates a random set of images in its “sleep” (in the mode of randomizing, if you will) and then “wakes up,” reviewing a reality through input to the computer and eliminates more radical images. It then returns to randomize on closely analogous images. The computer has been shown to become “more learned” in the sense of developing a more and more accurate image of reality.¹⁷⁴ Considering how babies begin to talk, such models sound logical empirical and theoretical.

This suggests that in twenty years, we may have biological blobs composed of DNA molecules functioning as primitive computers off of

174. See Y.S. Abu-Mostafa, *Machines that Learn from Hints*, SCI. AM, Apr. 1995, at 64 (indicating that machine learning is an area of tremendous technological growth).

microelectronic inputs. This revolutionary research illustrates the danger of too quickly concluding a technology is patentable, of allowing too broad a means clause, and of assuming that a formula or method has only narrow applicability. Perhaps a patent granted in five years on a DNA molecule will bar the use of that molecule in twenty years in broader contexts for a different, and as yet unimagined, function. A patent of a DNA sequence directed to a use or function with that use or function being stated as a limitation in the claim would remove this danger.

B. As Software Development Becomes More Industrialized and Less Academic, the Timing for Patentability of Software Is Right

The recent PTO Guidelines deal with the problem which occasioned them: the patentability of program storage devices. That is but one of four paradigms this article discusses. As suggested in *Patenting Mathematical Algorithms*,¹⁷⁵ the four paradigms of current interest are: (a) claims to pure computational methods, (b) claims to programmed apparatus, (c) claims to data structures, and (d) claims to program storage devices. The MPEP and the PTO Guidelines should go further in allowing patents on these paradigms but focus on restricting the claims to particular functions and force better disclosure.

Previously, most research in innovative program algorithms to improve existing problem solutions occurred in an academic setting, where there is pressure to publish. However, with the profit now to be achieved in the computer and computer software business, much research and development is privately funded, and the incentive to place the information and knowledge in the public domain is much less or even non-existent. Why should a car manufacturer with an innovative manufacturing solution involving a software algorithm yield any part of that knowledge to the public if no monopoly can be received? While the business method of the manufacturing solution should not be patentable, surely the technology of the software program for that particular manufacturing solution ought to be patentable as a new and useful process.

Allowing a patent monopoly for software focused on a particular function whose functionality is a limitation on the scope of the monopoly balances the public desire for disclosure against the inventor's reward of a limited monopoly. As computer technology unfolds in an exciting way, the grant of a limited monopoly in return for disclosure should, as it has in many other fields, spur other inventors onto new dis-

175. *Patenting Mathematical Algorithms*, *supra* note 4, § 6.0 (delineating differing courts' views on the patentability of algorithm-based material).

closures and new developments without undue stifling of innovation because of an overbroad monopoly. If the seventeen year monopoly is the concern, the PTO should recommend to Congress that the term of patents in computer implemented technology be more limited in time before the early inventors receive a long term and the later ones arbitrarily receive a shortened term.

The framers of the Constitution, could not foresee the modern digital computer. They likely would not have intended all software technology simply to be dumped into the classification of abstract ideas, laws of nature, and natural phenomena¹⁷⁶ to be forever excluded from the patent area. Therefore, we can surmise that some patentability would have been allowed. Analyzing the claims as method steps and requiring functional limitations on software patents reconciles competing patent policies and interests in this area of technology where the course of future development is not as yet well understood.

176. See *Diamond v. Diehr*, 450 U.S. 175, 185 (1981).