UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

MOBILEIRON, INC.,

Petitioner,

v.

BLACKBERRY, LTD.,

Patent Owner.

IPR2020-01741
U.S. Patent No. 7,372,961
Issue Date: May 13, 2008
Title: METHOD OF PUBLIC KEY GENERATION

DECLARATION OF MICHAEL T. GOODRICH, PH.D. IN SUPPORT OF PETITION FOR *INTER PARTES* REVIEW OF U.S. PATENT NO. 7,372,961

TABLE OF CONTENTS

I. QUALIFICATIONS 1
II. MATERIALS CONSIDERED 8
III. LEGAL STANDARDS 12
A. Prior Art 12
B. Claim Construction 12
C. Anticipation 15
D. Obviousness 16
IV. PERSON OF ORDINARY SKILL IN THE ART 23
V. SUMMARY OF OPINIONS 24
VI. TECHNOLOGICAL BACKGROUND 24
A. Applied Cryptography and Cryptographic Keys 25
B. Digital Signatures 29
C. Numbers, Strings, and Duality 31
D. Modular Arithmetic 32
E. Prime Numbers, Groups, and their Orders 35
F. Hash Functions 39
G. Random Number Generators (RNGs) 41
H. Rejection Sampling 44
VII. PRIOR ART REFERENCES 50
A. U.S. Patent 6,697,946 ("Miyaji") (Ex. 1005) 51
B. Bellare (Ex. 1006) 54
C. LEDA (Es. 1007) 56
D. Facebook's Prior Art 59
VIII. THE '961 PATENT 62
A. Summary of the '961 Patent 62
B. The File History for the '961 Patent 69
C. The Challenged Claims 74

D. Proposed Claim Constructions for the '961 Patent 75
IX. ANTICIPATION 80
A. Miyaji 80
i. Claims 1[a], 15[a], and 23[a] 81
ii. [23][b]: "an arithmetic processor for" 91
iii. 1[b], 15[b], 23[c]: "generating a seed value SV from a random number generator;" 92
iv. 1[c], 15[c]. 23[d]:"performing a hash function H( ) on said seed value SV to provide an output H(SV);" 93
v. 1[d], 15[d], 23[e]: "determining whether said output H(SV) is less than said order q prior to reducing mod q;" 95
vi. 1[e], 15[f], 23[f]: "accepting said output H(SV) for use as said key k if the value of said output is less than said order q;" 97
vii. 1[f], 15[f], 23[g]: "rejecting said output H(SV) as said key if said value is not less than said order q;" 98
viii. 1[g], 15[g], 23[h]: "if said output H(SV) is rejected, repeating said method; and" 99
ix. 1[h], 15[h], 23[i]: "if said output H(SV) is accepted, providing said key k for use in performing said cryptographic function, wherein said key k is equal to said output H(SV)." 100
x. Claims 2, 16, and 24 102
xi. Claims 3, 17, and 25 104
xii. Claims 4, 18, and 26 104
B. OBVIOUSNESS 106
i. Miyaji in View of Ordinary Skill in the Art 106
ii. Claims 1, 15, and 23 107
iii. Claims 2, 16, and 24 109
iv. Claims 3, 17, and 25 112
v. Claims 4, 18, and 26 113
vi. Claims 5, 19, and 27 118

vii. Claims 6, 20, and 28 121
C. Motivation to Combine 122
D. Obviousness by Bellare in View of LEDA 124
i. Claims 1, 15, and 23 125
ii. Claims 2, 16, and 24 147
iii. Claims 3, 17, and 25 148
iv. Claims 4, 18, and 26 148
v. Claim 5, 19, and 27 151
vi. Claims 6, 20, and 28 152
E. Motivation to combine 153
F. Alleged Secondary Considerations of Non-Obviousness 159
i. Alleged commercial success 160
ii. Alleged long felt but unresolved need 162
iii. Alleged failure of others 163
iv. Alleged copying of the invention by others....166

I, Michael T. Goodrich, Ph.D., declare and state as follows:

I. QUALIFICATIONS

1. I make this Declaration based upon my own personal knowledge, information, and belief, and I would and could competently testify to the matters set forth in this Declaration if called upon to do so.

2. Attached hereto as **Appendix A** is a true and correct copy of my Curriculum Vitae (CV). I received a Bachelor of Arts ("BA") degree in Mathematics and Computer Science from Calvin University in 1983 and a Ph.D. in Computer Science from Purdue University in 1987.

3. I am a Distinguished Professor in the Department of Computer Science at the University of California, Irvine, where I have been a faculty member since 2001. The Distinguished Professor title at University of California, Irvine is a campus-level distinction reserved for above-scale faculty who have achieved the highest levels of scholarship over the course of their careers and have earned national and international distinctions and honors of the highest level. Prior to working at the University of California, Irvine, I was a professor in the Department of Computer Science at the Johns Hopkins University from 1987-2001.

4. I have authored and coauthored over 300 publications, including several widely adopted books, such as *Introduction to Computer Security* and

Algorithm Design and Applications. My research includes contributions to data structures and algorithms, information security and privacy, networking, graph algorithms, computational geometry, distributed and parallel algorithms, and cloud security. For example, I have published research articles on topics in applied cryptography, network security, authentication and authorization, certificate revocation, forensics, location security, and certified email. Using the indexing scheme of my CV, examples of such publications include the publications corresponding to B-10, Ch-9, J-63, J-64, J-66, J-71, J-78, C-83, C-85, C-87, C-89, C-90, C-91, C-94, C-98, C-100, C-101, C-103, C-104, C-108, C-112, C-113, C-115, C-117, C-118, C-122, C-125, C-131, C-133, C-139, C-145, C-148, C-150, C-151, C-155, C-165, C-166, C-167, C-168, C-181, C-191, C-192, C-193, C-199, C-202, C-207, C-210, and C-216.

5. In addition, I have consulting experience in matters involving algorithms, cryptography, machine learning, digital rights management, computer security, networking, software, and storage technologies. I have consulted as a technical expert and expert witness on matters related to technologies relevant to the '961 Patent. For example, I was a technical expert and deponent and testified at trial regarding digital rights management and applied cryptography, including digital signatures, hash functions, and encryption/decryption in ContentGuard Holdings v.

Amazon, Apple, HTC, Samsung, Huawei, BlackBerry, and Motorola (Google). I have also consulted as a technical expert and deponent in IPRs and district court cases regarding network security, intrusion detection, and encryption/decryption in Finjan Inc. v. Blue Coat, Juniper Networks, Palo Alto Networks, Cisco Systems, Qualys, Rapid7, and SonicWall. In addition, I was a technical expert and deponent regarding encryption/decryption, public-key cryptography, key exchange, and key establishment in Philips v. Acer, ASUS, HTC, Southern Telecom, Visual Land, Zowee Marketing, Shenzen, YiFang, EFun, and Microsoft.

6. I received a number of awards and recognitions for my research, teaching, and service to the community. I am a Fellow of the American Association for the Advancement of Science (AAAS), a Fulbright Scholar, a Fellow of the institute of Electrical and Electronics Engineers (IEEE), and a Fellow of the Association for Computing Machinery (ACM). I am a foreign member of the Royal Danish Academy of Science and Letters. I am also a recipient of the IEEE Computer Society Technical Achievement Award, "for outstanding contributions to the design of parallel and distributed algorithms for fundamental combinatorial and geometric problems." I was named an ACM Distinguished Scientist in 2006. In terms of my teaching recognitions, I am a recipient of the Pond Award for Excellence in Undergraduate Teaching, as well as several Oraculum Awards for Excellence in

Teaching, from the Johns Hopkins University. I have also received a Chancellor's Award for Excellence in Fostering Undergraduate Research at the University of California, Irvine. In terms of my service recognitions, I received an ACM Recognition of Service Award in 1996 for my work on the committee establishing a Federated Computer Research Conference, which every 3-to-4 years brings together researchers across multiple subfields in Computer Science, including theory, software, and hardware, to share their latest breakthroughs.

7. I am familiar with applied cryptography, cryptographic theory, computer security, malware techniques, intrusion detection, virus detection, and anti-virus software systems and algorithms that were developed before and existed as of December 27, 2000, at the time the inventors of U.S. Patent No. 7,372,961 ("the '961 patent") (**Ex. 1001**) filed their underlying Canadian Patent Application. For example, I am particularly familiar with the applications, techniques, systems and algorithms related to applied cryptography, including encryption/decryption, hash functions, modular arithmetic, digital signatures, and random number generation.

8. My study of computer security topics began in the 1980s as an undergraduate student and continued in graduate school, where my Ph.D. research involved the study of computer system components operating in parallel. My study

and interest in computer security continued after I received my Ph.D., as detailed above and in my CV. In addition, I have reviewed and evaluated research papers on computer security, including applied cryptography, beginning with work as an associate editor for Journal of Computer & System Sciences, as well as my service on program committees of peer reviewed Computer Science conferences, including the Conference on Electronic Publishing and the Information Superhighway and the ACM Symposium on Theory of Computing, the latter for which I chaired and edited the conference proceedings in 1994.

9. My research has been supported by the Defense Advanced Research Projects Agency (DARPA), the National Science Foundation (NSF), the Office of Naval Research (ONR), the Army Research Office (ARO), and the National Security Agency (NSA). For example, I received several grants from the NSA to study computer security and applied cryptography topics, starting in 1998, as well as a $1.5 million grant from DARPA in 2000 to study scalable computer security and applied cryptography topics related to the dynamic coalitions that occur in military and humanitarian missions where trustworthy parties must interact with semi-trusted or untrusted partners.

10. I am a co-inventor on several U.S. patents, including U.S. Patent No. 7,257,711, "Efficient Authenticated Dictionaries with Skip Lists and Commutative

Hashing," which discloses secure distributed data authentication schemes based on cryptographic hash functions and digital signatures; U.S. Patent No. 7,299,219, "High Refresh-Rate Retrieval of Freshly Published Content using Distributed Crawling," which discloses a technology for quickly retrieving website data that can change frequently, so as to be stored in a search engine; U.S. No. Patent 8,681,145, "Attribute Transfer Between Computer Models Including Identifying Isomorphic Regions in Polygonal Meshes," which teaches how to map one mesh-based computer model to another; and U.S. Patent No. 9,152,716, "Techniques for Verifying Search Results Over a Distributed Collection," which discloses a system for searching the Internet so as to produce cryptographically verifiable search results that can be produced by a search engine.

11. I have taught courses at the Johns Hopkins University, Brown University, and the University of California, Irvine, at both the undergraduate and graduate levels. Topics of my courses have included computer security, algorithms, data structures, networking, algorithm engineering, computational geometry, and parallel processing. In addition, I have mentored 22 Ph.D. students over the years, who have written their Ph.D. theses on topics in algorithms, data structures, networking, parallel processing, applied cryptography, and computer security and privacy.

12. I have served as an editor on several technical journals, including Computational Geometry: Theory and Applications, the Journal of Computer & System Sciences, the Journal of Graph Algorithms and Applications, the International Journal of Computational Geometry & Applications, and Information Processing Letters. I have also served on many program committees (PCs) for top conferences and workshops in Computer Science, including serving as PC chair in several instances. Examples include ACM Symposium on Computational Geometry (SoCG), ACM Symposium on Theory of Computing (STOC), Workshop/Symposium on Algorithms and Data Structures (WADS), Algorithm Engineering and Experimentation (ALENEX, which I co-founded with Dr. Catherine McGeoch in 1999), IEEE Symposium on Foundations of Computer Science (FOCS), ACM-SIAM Symposium on Discrete Algorithms (SODA), International Symposium on Graph Drawing (GD), International Colloquium on Automata, Languages, and Programming (ICALP), ACM Conference on Computer and Communications Security (CCS), European Symposium on Algorithms (ESA), IEEE International Parallel and Distributed Processing Symposium (IPDPS), ACM Symposium on Parallel Algorithms and Architectures (SPAA), ACM Symposium on Advances in Geographic Information Systems (GIS), IEEE Symposium on Security and Privacy (S&P), IEEE International Conference on Big Data, IEEE

International Conference on Data Engineering (ICDE), and International Symposium on Algorithms and Computation (ISAAC).

II. MATERIALS CONSIDERED

13. The analysis that I provide in this Declaration is based on my education and experience in the field of computer security and encryption, as well as the documents I have considered. These documents include the '961 patent [**Ex. 1001**] and its prosecution history [**Ex. 1003**]. The '961 patent states on its face that it issued from an application filed on December 26, 2001, and claims priority to Canadian Patent Application No. 2329590, filed on December 27, 2000. For the purposes of this Declaration, I have assumed December 27, 2000 as the priority date for the '961 patent. I reserve the right, however, to show that the prior art anticipates and/or renders obvious various claims of the '961 patent based on an earlier or later priority date, if the facts warrant as much.

14. I have considered and/or cited to the following documents in my analysis, with cross-references to Exhibits used with the Petition for *Inter Partes* Review:

Exhibit No.	Description of Document
1001	U.S. Patent No. 7,372,961 B2 to Scott A. Vanstone et al. (filed December 26, 2001, issued May 13, 2008)
	Intentionally left blank

Exhibit No.	Description of Document
1003	Prosecution History for U.S. Patent Application No. 10/025,924, which issued as U.S. Patent No. 7,372,961
1004	Federal Information Processing (FIPS) Publication 186, *Digital Signature Standard (DSS)* (May 19, 1994)
1005	U.S. Patent 6,697,946 to Miyaji (PCT Application filed January 28, 1997, PCT Application published July 30, 1998, U.S. Patent issued February 24, 2004)
1006	"'Pseudo-random' Number Generation Within Cryptographic Algorithms: The DDS case," by Mihir Bellare, et al. (published in Proceedings of the Annual International Cryptology Conference (CRYPTO), pp. 277-291, Springer, 1997, as a part of the *Lecture Notes in Computer Science* (*LNCS*) book series, volume 129).
1007	Excerpts from *LEDA: a Platform for Combinatorial and Geometric Computing*, by Kurt Mehlhorn & Stefan Näher, Cambridge University Press, 1999
1008	Original Complaint filed April 27, 2020 in *MobileIron. Inc. v. Blackberry Corp., et al.*, Case No. 3:20-cv-02877 (N.D. Cal.)
1009	First Amended Complaint filed June 29, 2020, in *MobileIron. Inc. v. Blackberry Corp., et al.*, Case No. 3:20-cv-02877 (N.D. Cal.)
1010	First Amended Complaint filed in *BlackBerry Ltd. v. Facebook, Inc. et al.*, Case No. 2:18-cv-01844-GW-KS (C.D. Cal.), Facebook's Exhibit 1033 from *Facebook, Inc., et al. v. BlackBerry Ltd.*, IPR2019-00923 (PTAB)
1011	Excerpts from Menezes et al., Handbook of Applied Cryptography (1997)
1012	Dkt. Entry 157, *Corrected* Final Ruling on Claim Construction/*Markman* Hearing, filed April 5, 2019, entered April 11, 2019, in *BlackBerry Ltd. v. Facebook, Inc. et al.*, Case No. 2:18-cv-01844-GW-KS (C.D. Cal.)
1013	Dkt. Entry 652, Civil Minutes entered February 13, 2020, in *BlackBerry Ltd. v. Facebook, Inc. et al.*, Case No. 2:18-cv-01844-GW-KS (C.D. Cal.)

Exhibit No.	Description of Document
1014	Dkt. Entry 655, Sealed Minutes of Motion Hearing held February 27, 2020, before Judge George H. Wu, in *BlackBerry Ltd. v. Facebook, Inc. et al.*, Case No. 2:18-cv-01844-GW-KS (C.D. Cal.)
1015	Dkt. Entry 656, Joint Request To Vacate Pretrial Conference and Trial Date, filed March 19, 2020, in *BlackBerry Ltd. v. Facebook, Inc. et al.*, Case No. 2:18-cv-01844-GW-KS (C.D. Cal.)
1016	Dkt. Entry 657, Order dated March 24, 2020 Vacating Pretrial Conference and Trial Date, in *BlackBerry Ltd. v. Facebook, Inc. et al.*, Case No. 2:18-cv-01844-GW-KS (C.D. Cal.)
1017	Dkt. Entry 673, In Chambers – Order dated July 23, 2020, in *BlackBerry Ltd. v. Facebook, Inc. et al.*, Case No. 2:18-cv-01844-GW-KS (C.D. Cal.)
1018	Petition for *Inter Partes* Review, filed by Facebook, Inc., et al. in *Facebook, Inc., et al. v. BlackBerry Ltd.*, IPR2019-00923 (PTAB)
1019	Paper No. 14, Decision entered November 5, 2019, in *Facebook, Inc., et al. v. BlackBerry Ltd.*, IPR2019-00923 (PTAB Nov. 5, 2019)
1020	Excerpts from Schneier, Applied Cryptography (2d ed. 1996), Exhibit 1008 from *Facebook, Inc., et al. v. BlackBerry Ltd.*, IPR2019-00923 (PTAB)
1021	Rose, Re: "Card-shuffling" algorithms, USENET, sci.crypt, sciath h 10, 1993), Facebook's Exhibit 1006 from *Facebook, Inc., et al. v. BlackBerry Ltd.*, IPR2019-00923 (PTAB)
1022	Excerpts from Rao, Error Coding for Arithmetic Processors (1974), Facebook's Exhibit 1018 from *Facebook, Inc., et al. v. BlackBerry Ltd.*, IPR2019-00923 (PTAB)
1023	Excerpts from Floyd, Essentials of Data Processing (1987), Facebook's Exhibit 1033 from *Facebook, Inc., et al. v. BlackBerry Ltd.*, IPR2019-00923 (PTAB)
1024	Declaration of Daniele Micchiancio
1025	Affidavit of Elizabeth Rosenberg

Exhibit No.	Description of Document
1026	Declaration of Sylvia Hall-Ellis, Ph.D, in Support of Petitioner's Petition for *Inter Partes* Review
1027	Excerpts from Blackberry's Supplemental Responses and Objections to Facebook's Interrogatories, Facebook's Exhibit 1033 from *Facebook, Inc., et al. v. BlackBerry Ltd.*, IPR2019-00923 (PTAB)
1028	Declaration of Dr. Jonathan Katz, Facebook's Exhibit 1002 from *Facebook, Inc., et al. v. BlackBerry Ltd.*, IPR2019-00923 (PTAB)
1029	Patent Owner's Preliminary Response filed in *Facebook, Inc., et al. v. BlackBerry Ltd.*, IPR2019-00923 (PTAB)
1030	Declaration of Markus Jakobsson, Exhibit 2001 filed by Patent Owner in *Facebook, Inc., et al. v. BlackBerry Ltd.*, IPR2019-00923 (PTAB)
1031	Excerpts from *Microsoft Computer Dictionary, Fourth Edition,* 1999
1032	Federal Information Processing (FIPS) Publication 186-1, *Digital Signature Standard (DSS)* (December 15, 1998)
1033	Federal Information Processing (FIPS) Publication 186-2, *Digital Signature Standard (DSS)* (January 27, 2000)
1034	John von Neumann, *Various Techniques Used in Connection with Random Digits*, Summary by G.E. Forsythe, National Bureau of Standards Applied Math Series, 12 (1951), pp 36-38, reprinted in von Neumann's Collected Works, 5 (1963), Pergamon Press pp 768-770, Facebook's Exhibit 1017 from *Facebook, Inc., et al. v. BlackBerry Ltd.*, IPR2019-00923 (PTAB)
1035	Excerpts from M.T. Goodrich and R. Tamassia, *Algorithm Design and Analysis,* Wiley, 2015
1036	Federal Information Processing (FIPS) Publication 186-3, *Digital Signature Standard (DSS)* (June 2009), Patent Owner's Exhibit 2003 from *Facebook, Inc., et al. v. BlackBerry Ltd.*, IPR2019-00923 (PTAB)

15. In addition to these materials, I may consider additional documents and information in forming any supplemental opinions.

III. LEGAL STANDARDS

16. I am not an attorney. For purposes of this declaration, I have been informed about certain aspects of the law that are relevant to my analysis and opinions, as set forth below.

A. Prior Art

17. I understand that the prior art to the '961 patent includes patents and printed publications in the relevant art that predate the '961 patent's presumed December 27, 2000 priority date. As I explained previously, I have been instructed to assume for purposes of my analysis that December 27, 2000 is the earliest relevant date for determining what is "prior art." In other words, I should consider as "prior art" anything publicly available prior to December 27, 2000. I further understand that, for purposes of this proceeding in the United States Patent Trial and Appeal Board, only patents and documents that have the legal status of a "printed publication" may be relied on as prior art.

B. Claim Construction

18. I understand that under legal principles, claim terms are generally given their ordinary and customary meaning, which is the meaning that the term would

have to a person of ordinary skill in the art (POSA) in question at the time of the invention, i.e., as of the effective filing date of the patent application. I further understand that the POSA is deemed to read the claim term not only in the context of the particular claim in which a claim term appears, but in the context of the entire patent, including the specification.

19. I am informed by counsel that the patent specification, under the legal principles, has been described as the single best guide to the meaning of a claim term, and is thus highly relevant to the interpretation of claim terms. And I understand for terms that do not have a customary meaning within the art, the specification usually supplies the best context of understanding the meaning of those terms.

20. I am further informed by counsel that other claims of the patent in question, both asserted and unasserted, can be valuable sources of information as to the meaning of a claim term. Because the claim terms are normally used consistently throughout the patent, the usage of a term in one claim can often illuminate the meaning of the same term in other claims. Differences among claims can also be a useful guide in understanding the meaning of particular claim terms.

21. I understand that the prosecution history can further inform the meaning of the claim language by demonstrating how the inventors understood the invention

and whether the inventors limited the invention in the course of prosecution, making the claim scope narrower than it otherwise would be. Extrinsic evidence may also be consulted in construing the claim terms, such as my expert testimony.

22. I have been informed by counsel that, in IPR proceedings, a claim of a patent shall be construed using the same claim construction standard that would be used to construe the claim in a civil action filed in a U.S. district court (which I understand is called the "*Phillips*" claim construction standard), including construing the claim in accordance with the ordinary and customary meaning of such claim as understood by one of ordinary skill in the art and the prosecution history pertaining to the patent.

23. I have been instructed by counsel to apply the *Phillips* claim construction standard for purposes of interpreting the claims in this proceeding, to the extent they require an explicit construction. The description of the legal principles set forth above thus provides my understanding of the *Phillips* standard as provided to me by counsel.

24. I understand that some claims are independent, and that these claims are complete by themselves. Other claims refer to these independent claims and are "dependent" from those independent claims. The dependent claims include all of the limitations of the claims on which they depend.

C. Anticipation

25. I understand that to anticipate a patent claim under 35 U.S.C. § 102, a single asserted prior art reference must disclose each and every element of the claimed invention, either explicitly, implicitly, or inherently, to a POSA. There must be no difference between the claimed invention and the disclosure of the alleged prior art reference as viewed from the perspective of ta POSA. Also, I understand that in order for a reference to be an anticipating reference, it must describe the claimed subject matter with sufficient clarity to establish that the subject matter existed and that its existence was recognized by persons of ordinary skill in the field of the invention. In addition, I understand that in order to establish that an element of a claim is "inherent" in the disclosure of an asserted prior art reference, extrinsic evidence (or the evidence outside the four corners of the asserted prior art reference) must make clear that the missing element is necessarily found in the prior art, and that it would be recognized as necessarily present by persons of ordinary skill in the relevant field.

26. In my opinions below, when I say that a POSA would understand, readily understand, or recognize that an element or aspect of a claim is disclosed by a reference, I mean that the element or aspect of the claim is disclosed explicitly to a POSA.

D. Obviousness

27. I am also informed and understand that even though a prior art reference does not fully anticipate a claim of a patent, a claim may, nonetheless, be rendered obvious to a POSA if the differences between the subject matter set forth in the patent claim and the prior art are such that the subject matter as a whole of the claim would have been obvious at the time the claimed invention was made.

28. I understand that obviousness is a determination of law based on various underlying determinations of fact. In particular, these underlying factual determinations include (1) the scope and content of the prior art; (2) the level of ordinary skill in the art at the time the claimed invention was made; (3) the differences between the claimed invention and the prior art; and (4) the extent of any proffered objective indicia of nonobviousness. I understand that the objective indicia which may be considered in such an analysis include commercial success of the patented invention (including evidence of industry recognition or awards), whether the invention fills a long-felt but unsolved need in the field, the failure of others to arrive at the invention, industry acquiescence and recognition, initial skepticism of others in the field, whether the inventors proceeded in a direction contrary to the accepted wisdom of those of ordinary skill in the art, and the taking of licenses under the patent by others, among other factors.

29. I understand that to ascertain the scope and content of the prior art, it is necessary to first examine the field of the inventor's endeavor and the particular problem for which the invention was made. The relevant prior art includes prior art in the field of the invention, and also prior art from other fields that a POSA would look to when attempting to solve the problem.

30. I understand that a determination of obviousness cannot be based on the hindsight combination of components selectively culled from the prior art to fit the parameters of the patented invention. Instead, it is my understanding that in order to render a patent claim invalid as being obvious from a combination of references, there must be some evidence within the prior art as a whole to suggest the desirability, and thus the obviousness, of making the combination in a way that would produce the patented invention.

31. I further understand that in an obviousness analysis, neither the motivation nor the purpose of the patentee dictates. What is important is whether there existed at the time of the invention a known problem for which there was an obvious solution encompassed by the patent's claims.

32. I further understand that a single reference can support a finding of obviousness if it would been obvious to modify that reference to arrive at the patented invention.

33. In addition, it is my understanding that in order to find a patent claim invalid for obviousness, there must be a finding that each element in each limitation of the patent claim is disclosed, taught, or suggested by the asserted combination of prior art references or elsewhere in the relevant prior art. I understand, however, that a patent claim composed of several elements is not proved obvious merely by demonstrating that each of its elements was, independently, known in the prior art. But multiple prior art references or elements may, in some circumstances, be combined to render a patent claim obvious. I understand that I should consider whether there is an "apparent reason" or motivation to combine the prior art references or elements in the way the patent claims. To determine whether such an "apparent reason" or motivation exists to combine the prior art references or elements in the way a patent claims, it will often be necessary to look to the interrelated teaching of multiple prior art references, to the effects of demands known to the design community or present in the marketplace, and to the background knowledge possessed by a POSA.

34. I also understand that when the prior art "teaches away" from combining prior art references or certain known elements, discovery of a successful means of combining them is more likely to be nonobvious. A prior art reference may be said to "teach away" from a patent when a POSA, upon reading the

reference, would be discouraged from following the path set out in the patent or would be led in a direction divergent from the path that was taken by the patent. Additionally, a prior art reference may "teach away" from a claimed invention when substituting an element within that prior art reference for a claim element would render the claimed invention inoperable.

35. I am informed and understand that a patent for a combination which only recites old elements with no change in their respective functions withdraws what is already known into the field of its monopoly and diminishes the resources available to a POSA. When a patent claims a structure already known in the prior art that is altered by mere substitution of one element for another known in the field, the combination must do more than yield a predictable result. I am informed and understand that the corollary to this principle is that when the prior art teaches away from combining known elements, discovery of a successful means of combining them is more likely to be nonobvious. Thus, the question to be answered is whether someone reading the prior art would be discouraged from following the path taken by the inventor.

36. I am further informed and understand that when an element is available in one field of endeavor, design incentives and other market forces can prompt variations of it, either in the same field or a different one. If a POSA can implement

a predictable variation of that available element, § 103 likely renders the invention obvious. For the same reason, I am informed and understand that if a technique has been used to improve one device or process, and a POSA would recognize that such technique would improve similar devices or processes in the same way, using the technique is obvious unless its actual application is beyond his or her skill. Following these principles often requires one to look to interrelated teachings of multiple patents; the effects of demands known to the design community present in the marketplace; and the background knowledge possessed by a person having ordinary skill in the art, all in order to determine whether there was an apparent reason to combine the known element in the manner claimed by the patent at issue. I am further informed and understand that the analysis need not seek out precise teachings directed to the specific subject matter of the challenged claim, because one can take account of the inferences and creative steps that a POSA would employ.

37. I am further informed and understand that although the use of the "teaching, suggestion or motivation" test for combining references has not been completely rejected, the obviousness analysis cannot be confined by a formalistic conception of the words teaching, suggestion, and motivation, or by overemphasis on the importance of published articles and the explicit content of issued patents. I

am informed and understand that in many fields there is little discussion of obvious techniques or combinations, and it is often the case that market demand, rather than scientific literature, drive design trends. Under the correct analysis, any need or problem known in the field of endeavor at the time of the invention and addressed by the patent can provide a reason for combining the elements in the manner claimed. Finally, I am informed and understand that common sense teaches that familiar items may have obvious uses beyond their primary purposes and, in many cases, a POSA will be able to fit the teachings of multiple patents together like pieces of a puzzle.

38. I also am informed that I may consider the following factors to determine whether there are any secondary considerations of non-obviousness that may shed light on the obviousness or not of the claimed inventions: (a) Whether the invention was commercially successful as a result of the merits of the claimed invention (rather than the result of design needs or market-pressure advertising or similar activities); (b) whether the invention satisfied a long-felt need; (c) whether others had tried and failed to make the invention; (d) whether others invented the invention at roughly the same time; (e) whether others copied the invention; (f) whether there were changes or related technologies or market needs contemporaneous with the invention; (g) whether the invention achieved

unexpected results; (h) whether others in the field praised the invention; (i) whether persons having ordinary skill in the art of the invention expressed surprise or disbelief regarding the invention; (j) whether others sought or obtained rights to the patent from the patent holder; and (k) whether the inventor proceeded contrary to accepted wisdom in the field.

39. I am also informed that, consistent with what I state above, the following exemplary rationales all may support a conclusion of obviousness: (a) combining prior art elements according to known methods to yield predictable results; (b) simple substitution of one known element for another to obtain predictable results; (c) use of known technique to improve similar devices (methods, or products) in the same way; (d) applying a known technique to a known device (method, or product) ready for improvement to yield predictable results; (e) "obvious to try" – choosing from a finite number of identified, predictable solutions, with a reasonable expectation of success; (f) known work in one field of endeavor may prompt variations of it for use in either the same field or a different one based on design incentives or other market forces if the variations are predictable to one of ordinary skill in the art; or (g) Some teaching, suggestion, or motivation in the prior art that would have led one of ordinary skill to modify the prior art reference or to combine prior art reference teachings to arrive at the claimed invention.

40. Therefore, I will analyze the prior art applied in Petitioners' proposed grounds using this framework.

IV. PERSON OF ORDINARY SKILL IN THE ART

41. I understand that the factors to be considered in determining the level of ordinary skill in the art to be: (1) the educational level of active workers in the field, including the named inventors of the patent; (2) the type of problems encountered in the art; (3) prior art solutions to those problems; (4) the rapidity with which innovations are made; and (5) the sophistication of the technology in the art. As mentioned above, I further understand that the earliest priority date for the '961 is December 27, 2000.

42. It is my opinion that the POSA in the field of the '961 patent would be someone with a bachelor's degree in computer science, mathematics or applied mathematics, or a related field, and (1) two or more years of experience in applied cryptography or computer security software engineering, and/or (2) an advanced degree in computer science or related field.

43. Based on my training and experience, I believe that I am a person of greater-than-ordinary skill in the relevant art and, as of December 27, 2000, was a person of at least ordinary skill in the relevant art. These qualifications permit me

to give an opinion about the qualifications of one of ordinary skill at the time of the invention.

V. SUMMARY OF OPINIONS

44. In summary, my opinions contained in this declaration are as follows:

- Claims 1-4, 15-18, and 23-26 of the ’961 patent are anticipated by Miyaji (**Ex. 1005**, or “Miyaji”).
- Claims 1-6, 15-20, and 23-28 of the ’961 patent are rendered obvious by Miyaji, e.g., in light of the admitted knowledge of a POSA.
- Claims 1-6, 15-20, and 23-28 of the ’961 patent are rendered obvious by Bellare (**Ex. 1006**, or “Bellare”) in light of LEDA (**Ex. 1007**, or “LEDA”).
- The evidence I have examined for alleged secondary considerations of non-obviousness do not overcome the above obviousness opinions.

45. In the sections that follow, I provide facts and reasons to support the above opinions.

VI. TECHNOLOGICAL BACKGROUND

46. In this section I review some of the relevant technologies that relate to the ’961 patent.

A. Applied Cryptography and Cryptographic Keys

47. The ’961 patent, entitled “Method of Public Key Generation,” purports to disclose a method for avoiding potential bias in the generation of a cryptographic key. ’961 patent, 3:1-3. The ’961 patent focuses primarily on a single aspect of cryptographic functions – the generation of the appropriate cryptographic key.

48. Thus, I briefly review applied cryptography, including cryptographic keys, and explain how generation of such keys can introduce biases that need to be minimized. In that regard, cryptographic keys are numbers or strings of bits used for encrypting or decrypting information to be transmitted.[1] Applied cryptography and cryptographic keys were concepts that were well-known to persons of skill in the art and well-documented as of December 27, 2000, the priority date for the ’961 patent.

49. Techniques for achieving confidentiality in communications have been around for a very long time, tracing their origins back to the time of the Caesars in ancient Rome. Moreover, techniques for achieving confidential communications

---

[1] *See, e.g., Microsoft Computer Dictionary, Fourth Edition,* 1999 (**Ex. 1031**) (“**key** *n.* In encryption and digital signatures, a string of bits used for encrypting and decrypting information to be transmitted. Encryption commonly relies on two different types of keys, a public key known to more than one person (say, both the sender and the receiver) and a private key known only to one person (typically, the sender).”) (emphasis in original).

were used in every U.S. conflict, including the Revolutionary War, the Civil War, and both World Wars.[2]

50. Cryptography is the study of mathematical techniques for performing functions related to information security, including confidentiality, as well as functions involving authentication, authorization, data integrity, validation, access control, certification, timestamping, verification, receipt, ownership, anonymity, privacy, revocation, commitment, and proof of knowledge.[3] For example, to achieve confidentiality, two parties, Alice and Bob, may share a secret number, called a "key," k, such that Alice can scramble a "plaintext" message, M, using the key k, to produce a "ciphertext," C, and send it to Bob, who can unscramble the ciphertext C using k to retrieve the message M. The scrambling step is called "encryption" and the unscrambling step is called "decryption." The goal of such an encryption/decryption scheme is that it should be difficult, if not impossible, for any third party to recover the plaintext message M from C without knowing the secret key, k.

51. The above example involving Alice and Bob is known as a "symmetric" key protocol, because the same key, k, is used for both encryption and decryption.

---

[2] *See, e.g.,* **Ex. 1011** ("Menezes") at pp. 1, 239, 242-243.
[3] *See, e.g.,* Menezes at p. 3.

A POSA would be aware of symmetric encryption/decryption schemes, such as the Data Encryption Standard (DES), which was published as an official Federal Information Processing Standard (FIPS) for the United States in 1977.[4] The requirement that both Alice and Bob possess the same secret key, k, poses a challenge, however, in that before Alice and Bob can engage in this symmetric encryption/decryption protocol, there needs to be a way for them to share the secret key, k. Historically, this had been done by physical means, such as sending a courier from Alice to Bob with the secret key, k; but in the digital age, such physical means are not always feasible.

52. In the 1970s, a way was devised to get around this problem, giving rise to "public-key cryptography," which would be well-understood by a POSA. As the '961 patent states: "The basic structure of public key cryptosystems is well known and has become ubiquitous with security in data communications systems." '961 patent at 1:10-12. In public-key cryptography, which is also known as "asymmetric encryption/decryption," a party, Bob, generates two keys—a public key, K_{pub}, and a private key, K_{priv}. Bob keeps the private key, K_{priv}, secret and does not share it

[4] *See, e.g.,* Data Encryption Standard, https://en.wikipedia.org/wiki/Data_Encryption_Standard, last visited September 16, 2020.

with anyone, but he shares the public key, K_{pub}, with anyone (he may even post it in a public space, such as his personal website). In such a public-key cryptosystem, Alice can encrypt a plaintext message, M, meant for Bob using Bob's public key, K_{pub}, and send him the resulting ciphertext, C. Bob can then decrypt the ciphertext, C, using his private key, K_{priv}, to retrieve the original message, M. In such a public-key cryptosystem, without knowing Bob's private key, K_{priv}, it is difficult, if not impossible, for a third party to decrypt the plaintext, M, from a ciphertext, C, encrypted with Bob's public key.

53. As the '961 patent acknowledges to be background knowledge of a POSA, public-key cryptosystems can be used for key agreement protocols, where Alice and Bob use a public-key cryptosystem to agree on a secret key, k, that they then share and can use for symmetric encryption/decryption. *See, e.g.,* '961 patent at 1:17-20 ("Similarly, the cryptosystems may be used for key agreement protocols where each party exponentiates the other party's public key with their own private key."). Some of these protocols employ the use of a pair of private keys and corresponding public keys, referred to as long term and short term or "ephemeral key" pairs. *See id.* at 1:38-42. The ephemeral private key is generated at the start of each session between a pair of correspondents, usually by a random number generator (or RNG). *Id.* at 1:42-44. The corresponding, ephemeral public key is

generated, and the resultant key pair used in one of a number of possible operations, like that described above involving Bob and Alice. *Id.* at 1:44-47. A long-term public key is utilized to authenticate the correspondent through multiple protocols. *Id.* at 1:47-49.

54. A POSA as of December 27, 2000 would also be aware of advantages of these and other prior art public-key schemes, like the ElGamal and Rivest-Shamir-Adleman ("RSA") schemes. *Id.* at 1:52-1:56; 2:23-29. Public-key cryptosystems tend to be slower than symmetric encryption/decryption protocols like DES. Using a public-key cryptosystem to agree on a secret shared key solves the problem of sharing a secret key between two parties without using physical means, which then allows them to use a fast symmetric encryption scheme like DSA for their subsequent communications.

B. Digital Signatures

55. Another functionality that the '961 Patent acknowledges as background knowledge of a POSA is that public-key cryptography may be used for digital signatures. *See id.* at 1:26-31. In this application, one party, Bob, can perform an operation that is a form of encryption[5] on a message, M, using his private key, K_{priv},

[5] In some digital signature schemes, Bob literally encrypts a message, M, using his private key as a way of digitally signing the message, M. In all digital signature

to produce a digital signature, S, such that anyone knowing Bob's public key can verify that S is a digital signature for M. Such schemes are designed so that it is difficult, if not impossible, for someone to forge a signature, S, for a message, M, without knowledge of Bob's private key. In fact, the '961 patent acknowledges this basic knowledge of the POSA as follows:

> Public key cryptosystems may also be used to sign messages to authenticate the author and/or the contents. In this case the sender will sign a message using his private key and a recipient can verify the message by applying the public key of the sender. If the received message and the recovered message correspond then the authenticity is verified.

'961 patent at 1:26-32.

56. Thus, in the context of digital signatures, a POSA would understand that signing a message involves a form of encryption using a private key as one component of the signature and in which the signature is verified through a form of decryption that also uses a public key as another component. *See*, *e.g.*, '961 patent, at 1:63-2:14.

57. As the '961 patent admits, a POSA would be well aware of popular digital signature schemes, including ElGamal, DSA, and another protocol known as

---

schemes, a signature verifier uses Bob's public key and a signature, S, for a message, M, in a derivation for the message, M.

ECDSA. *See, e.g.,* '961 Patent at 1:52-62 ("Some of the more popular protocols for signature are the ElGamal family of signature schemes such as the Digital Signature Algorithm or DSA") and 2:15-22 ("A similar signature protocol known as ECDSA may be used for elliptic curve cryptosystems"). All of these digital signature schemes require encryption using a private key and decryption employing a public key. For instance, the '961 patent notes: "The DSA algorithm utilizes both long term and ephemeral keys to generate a signature of the message." *Id.* at 1:54-56. "The DSA domain parameters are preselected," and, "[t]hey consist of a prime number p of a predetermined length, by way of example 1024 bits; a prime number q of a predetermined bit length, by way of example 160 bits, where q divides p−1; a generator α lying between 2 and p−1 and which satisfies the condition ($\alpha^q \bmod p)=1$, and; a cryptographic hash function H, such as SHA-1." *Id.* at 1:56-62.

C. Numbers, Strings, and Duality

58. Because character strings can also be treated as numbers, the '961 patent describes its functionality in terms of operations on numbers. For this overview, when I refer to "numbers" I am restricting my attention to integers, i.e., numbers without fractional parts, such as the numbers 0, 1, 2, 3, etc.

59. Integers are useful in cryptographic applications, because they can be represented as a string of bits -- that is, using the binary digits, 0 and 1. For example,

using two bits, we can represent 0 as 00, 1 as 01, 2 as 10, and 3 as 11. Using 3 bits we can represent 0 as 000, 1 as 001, 2 as 010, 3 as 011, 4 as 100, 5 as 101, 6 as 110, and 7 as 111. And so on. Thus, since any message, m, can also be represented as a string of bits, we can think of m itself as an integer and we can perform operations and functions on m by treating it as an integer. For example, the 8-bit binary string "01010000" can be viewed either as the integer 80 or the capital letter "P" (according to the ASCII[6] binary encoding standard). Indeed, a POSA would understand this practice of interpreting binary strings both as integers and strings of characters to be common in cryptography. Computer scientists use the term "duality" to refer to this concept of interpreting a binary string either as a number or a string of characters.

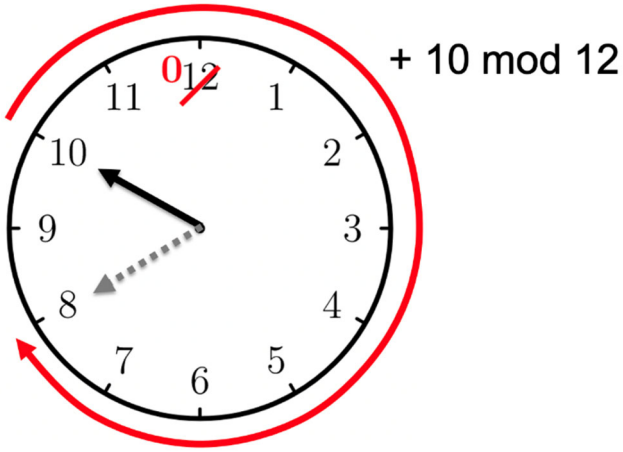
D. Modular Arithmetic

60. Using duality and viewing a binary string, x, as a number allows one to perform arithmetic operations on x, such as addition and multiplication, e.g., to encrypt or decrypt the string representation of x. For example, in modular arithmetic, the *modulo* operation, "x mod n," returns the remainder after dividing x by n. That is, x mod n = y if and only if x = a*x + y for some integer, a, and y is in the range from 0 to n-1, inclusive. Thus, 10 mod 3 is 1, 15 mod 9 is 6, and 30 mod

[6] American Standard Code for Information Interchange.

6 is 0. The process of performing a modulo operation is sometimes referred to as "reducing mod n" or a "reduction mod n."

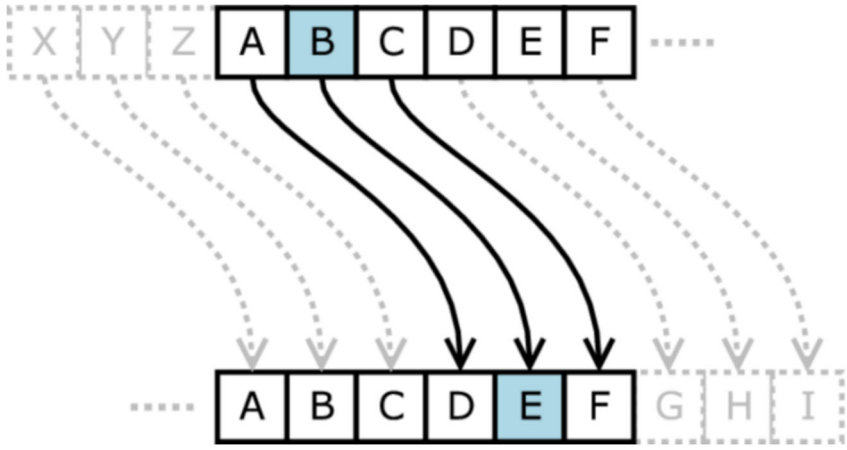
61. We can think of modular arithmetic as "clock arithmetic," because it is related to the way we tell time. For example, consider an analog clock where we replace the "12" with a "0." Then adding hours to a time, going clockwise around the clock, is addition mod 12. For instance, if a mother tells her teenage son to go to bed at 10:00 and to set his alarm to wake him up 10 hours later, he would calculate 10+10 mod 12 = 8 to realize his mother is asking him to set his alarm for 8:00:



62. Modular arithmetic is also useful for cryptography. For example, there is a famous encryption scheme that dates back to the time of the Caesars in Ancient Rome, which is known as "The Caesar Cipher,"[7] that is based on modular

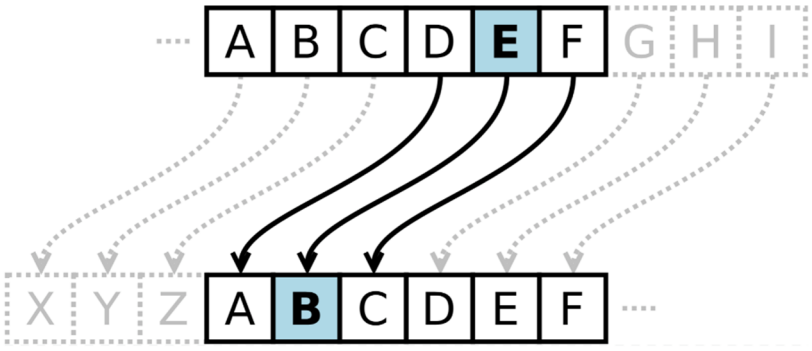
[7] *See, e.g.,* Menezes at p. 239.

arithmetic. In this scheme, one views each letter of the alphabet as a number, so 0 is A, 1 is B, 2 is C, and so on.[8] One encrypts a message by adding 3 mod 26 to each number (representing a letter) in a message. Another way to view this is that one substitutes each letter with the letter that comes 3 places after it in an alphabetical listing, wrapping around the end of the listing when one comes to the end. As in the clock example, modular arithmetic facilitates this "wrapping around" functionality. Thus, 0 (A) is encrypted as 0+3 mod 26 = 3 (D), 1 (B) is encrypted as 1+3 mod 26 = 4 (E), …, 23 (X) is encrypted as 23+3 mod 26 = 0 (A), 24 (Y) is encrypted as 24+3 mod 26 = 1 (B), and 25 (Z) is encrypted as 25+3 mod 26 = 2 (C). So, the message "ATTACK" is encrypted as "DWWDFN." Thus, encryption in the Caesar Cipher can be visualized as follows:



[8] Caesar used the Latin alphabet, of course, but I am using the English alphabet for illustrative purposes.

63. To decrypt a message, one simply subtracts 3 mod 26 from each number representing a letter in the message. Technically, there are no negative numbers in modular arithmetic, so "subtracting 3 mod 26" really means "adding 23 mod 26". For example, 4+23 mod 26 = 27 mod 26 = 1. Using the language of pure mathematics, we would say in this case that 23 is the "additive inverse" of 3 modulo 26, i.e., it acts like -3 in addition mod 26. Thus, decryption in the Caesar Cipher can be visualized as follows:



E. Prime Numbers, Groups, and their Orders

64. To explain how its cryptographic key is generated, the '961 patent introduces several other concepts from pure mathematics, including claim terms such as "group," "order" and "prime number," which the '961 patent presumes a POSA will already be familiar with. *See, e.g.,* '961 patent at 1:52-2:22, 3:64-4:10, and claims 1, 5, 15, 19, 23, and 27. Hence, I provide here a brief overview of some the most relevant of these concepts.

65. A number, p, is a *prime number* if the only numbers that divide p without remainder are 1 and p. For example, 2, 3, 17, and 59 are prime numbers, but 12 is not a prime number, since 6 divides 12 without a remainder. That is, 12 is not prime, because 12 mod 6 = 0. *See, e.g.,* Menezes at p. 64.

66. The set of integers, {0, 1, …, n-1}, is sometimes denoted as "$\mathbf{Z}_n$." *See, e.g.,* Menezes at p. 68. We can define the operations of addition, subtraction, and multiplication to all be modulo n, and in doing so we are guaranteed that the addition, subtraction, or multiplication of any two numbers in $\mathbf{Z}_n$, modulo n, will have its result in $\mathbf{Z}_n$. That is, $\mathbf{Z}_n$ is *closed* under addition, subtraction, and multiplication, modulo n. Also, in $\mathbf{Z}_n$, we define an *additive inverse* of a number, x, as a number, y, such that x+y (mod n) is 0. Similarly, we define a *multiplicative inverse* of a number, x, as a number, y, such that x*y (mod n) is 1. Every number in $\mathbf{Z}_n$ has an additive inverse, but it is not always the case that every non-zero number in $\mathbf{Z}_n$ has a multiplicative inverse. Indeed, a non-zero number, x, in $\mathbf{Z}_n$ has a multiplicative inverse if and only if the greatest common divisor ("gcd") of x and n is 1, that is, gcd(x,n)=1. *See, e.g.,* Menezes at p. 68.

67. A *group* is defined as a set, G, with an operation, "*," defined on the elements of G such that G is closed under the "*" operation, there is an *identity* element, "I" (that is, x*I=I*x=x), the "*" operation is associative (i.e.,

x*(y*z)=(x*y)*z), and every element in G has an inverse with respect to "*." The *order* of a group is the number of elements in the group. Thus, $\mathbf{Z}_n$ is a group with respect to the addition mod n ("+") operation (with I=0) and the order of $\mathbf{Z}_n$ is n. *See, e.g.,* Menezes at pp. 69, 75.

68. We define the *multiplicative group* of $\mathbf{Z}_n$, denoted $\mathbf{Z}^*_n$, with respect to the multiplication operation (and with I=1), to be the set of all non-zero numbers, x, in $\mathbf{Z}_n$ such that gcd(x,n)=1. *See, e.g.,* Menezes at p. 69. As with any group, the *order* of $\mathbf{Z}^*_n$ is the number of elements in $\mathbf{Z}^*_n$. For example, if n is a prime number, then the order of $\mathbf{Z}^*_n$ is n-1 (since 0 is not in $\mathbf{Z}^*_n$). For a prime number, p, we sometimes denote using "GF(p)" the set, $\mathbf{Z}_p$, with both of the operations addition and multiplication (mod p), i.e., GF(p) can be thought of as $\mathbf{Z}_p$ combined with $\mathbf{Z}^*_p$. Using the language of pure mathematics, GF(p) is a special kind of "*field*"[9] known as a "Galois field." *See, e.g.,* Menezes at p. 81.

69. We can define a (sub)group of $\mathbf{Z}^*_n$ by taking any element, a, in $\mathbf{Z}^*_n$ and considering all the numbers of the form, a^k (mod n), for k=0,1,2,…. As with any group, the *order* of this group is its number of (distinct) elements. An equivalent

[9] A field is a set with two operations, "+" and "*", such that the set is an additive group with respect to "+" and a multiplicative group with respect to "*". *See, e.g.,* Menezes at pp. 76-77.

way to define the *order* for any element, a in $\mathbf{Z}^*_n$, is the smallest positive number, k, such that $a^k \pmod n = 1$. For example, the following table shows the order of each number in $\mathbf{Z}^*_{21} = \{1, 2, 4, 5, 8, 10, 11, 13, 16, 17, 19, 20\}$:

$a \in \mathbb{Z}^*_{21}$	1	2	4	5	8	10	11	13	16	17	19	20
order of a	1	6	3	6	2	6	6	2	3	6	6	2

Menezes at p. 69.

70. Note that orders of the elements 4, 8, 13, 16, and 20 in $\mathbf{Z}^*_{21}$ are prime. This is no accident. By Cauchy's theorem,[10] if G is a finite group and q is a prime number dividing the order of G, then G contains an element of order q. For example, for each prime number, q, that divides (p-1), where p is a prime number, there is an element x in $\mathbf{Z}^*_p$ such that x has order q. That is, $x^q \pmod p = 1$. This fact is used, for example, in the establishment of the parameters for the Digital Signature Algorithm (DSA).

71. An algorithmic problem concerning numbers of the form "$g^k \pmod p$" that is generally considered to be computationally difficult, depending on the parameters g and p, is the *Discrete Logarithm* problem. *See, e.g.,* Menezes at p. 103. In this problem, one is given a value, $y=g^k \bmod p$, and told the values of g and

[10] *See, e.g.,* "Cauchy's theorem (group theory)," https://en.wikipedia.org/wiki/Cauchy%27s_theorem_(group_theory), last visited August 18, 2020.

p, but not k, and the problem is to compute k. There is to date no efficient way to solve this problem for all parameters g and p. This difficulty forms the basis of security for a number of cryptographic functions, including the DSA signature scheme. In order to enhance the security of any scheme, such as DSA, whose security is based on the Discrete Logarithm problem for a prime parameter, p, it is important that the prime p be chosen such that p-1 has at least one large prime factor. This is due to an effective algorithm for solving the Discrete Logarithm problem, due to Pohlig and Hellman,[11] when p-1 has only small prime factors. A POSA would be aware of this algorithm and would understand from it that the security of a cryptographic system based on the difficulty of the Discrete Logarithm problem p is therefore enhanced by choosing p so that p-1 has at least one large prime factor, q, and choosing g to have an order q where q is a prime number represented by a binary string with a significant number of bits, e.g., at least 160 bits.

F. Hash Functions

72. Applied cryptography also employs other mathematical operations to achieve various cryptographic functions, which would be well-known to a POSA as of December 27, 2000. For example, a "hash function," h(x), is a means to produce a relatively short digest or fingerprint for a message, M, such as by mapping a value,

[11] *See, e.g.,* Menezes at pp. 104, 107-109.

x, which could correspond to an entire message, to a 160-bit digest. *See, e.g.,* Menezes at p. 321. For example, one type of hash function, is a "***multiplicative hash function***," which is a hash function of the following form:

- $h(x) = (a*x + b) \bmod n$,

where a, b, and n, are positive integers (e.g., 160-bit numbers). Another type of hash function, which is known as an "***exponential hash function***" or "discrete logarithm hash function," is a hash function of the following form:

- $h(x) = g^x \bmod n$, for suitable parameters g and n.[12]

73. A POSA would also understand that the composition of two hash functions is itself a hash function. That is, if f( ) and g( ) are hash functions, then h(x) = f(g(x)) is also a hash function. This is because each hash function maps its input to an integer digest; hence, composing two hash functions maps the input of the first hash function to a digest of its input, which is then itself mapped to a digest by the second hash function, which by simple transitivity is a digest for the original input.

74. In cryptographic applications, sometimes one desires an additional property for a hash function, such that the result of performing a hash function, h(x),

[12] *See, e.g.,* Menezes at p. 329 (Example 9.17 in the chapter, "Hash Functions and Data Integrity").

on an input, x, appears to be random, so that it is difficult to determine the value of x by just knowing the value of h(x). Examples of hash functions, which would be known to a POSA as of December 27, 2000, that are considered to have this random "mixing" functionality, are hash functions known as "MD5"[13] and "SHA-1."[14] For example, SHA-1 is recommended as a hash function to use with the Digital Signature Algorithm, DSA.[15]

G. Random Number Generators (RNGs)

75. Cryptography relies heavily upon random numbers. A POSA as of December 27, 2000 would therefore be aware of random number generators (RNGs), which are methods for generating a sequence of numbers that have the properties of random numbers. Such random number generators typically output numbers that are uniformly distributed within a large, specified range. For example, if a random number generator generates 32-bit numbers, then it can generate values ranging from zero to 2^{32} minus 1, which is 4,294,967,295. Treating this extremely large number as a single integer provides many useful applications when developing cryptographic algorithms. Random number generators are used in many

[13] *See, e.g.,* Menezes at p. 347.
[14] *See, e.g.,* Menezes at pp. 348-349.
[15] *See, e.g.,* FIPS 186-1 (Ex. 1032) at p. 14.

cryptographic protocols, including the methods of the '961 patent. *See, e.g.,* '961 patent at 1:33-50, 2:23-46, 3:33-44, 4:6-52, and claims 1, 15, and 23. There are two types of random number generators—"truly" random number generators and "pseudo-random" number generators.

76. Just as its name implies, a truly random number generator will generate a sequence of truly random numbers, such as would come from rolling dice or tossing coins. Truly random number generators are ideal for cryptographic applications, because they are provably unpredictable. The challenge in real-world computing applications, however, is that truly random numbers are hard to come by. For example, even in settings where randomness is available, such as in sampling radio static or a radioactive source, gathering a big enough sample to achieve true randomness takes time and energy. Thus, a POSA would understand that in real-world computing applications, truly random number generators are not common.

77. A pseudo-random number generator, on the other hand, is a computational function that take as input a number, s, known as a "seed" and produces a sequence of numbers from that seed such that the output numbers appear to be random. The seed, s, could be truly random or it could, itself, come from a random number generator. Each step in a pseudo-random number generator is

deterministic; hence, its randomness depends on the initial seed, s, and the generator's ability to exploit that randomness to create a sequence of statistically random numbers.

78. A POSA as of December 27, 2000 would also know that one type of pseudo-random number generator repeatedly applies a hash function, such as a cryptographic hash function or a multiplicative hash function, to the output of the previous iteration, starting from a seed value, such as from user input or an output from a random number generator. For instance, repeatedly applying a multiplicative hash function to the output of the previous iteration is known as a "linear congruential generator." *See, e.g.,* Bellare (**Ex. 1006**), as I describe below. Further, years before the earliest priority date for the '961 patent, two of the inventors on the '961 patent described the well-known hash-based type of random number generator in their textbook, *Handbook on Applied Cryptography* (**Ex. 1011**), e.g., as follows:

(ii) Software-based generators

Designing a random bit generator in software is even more difficult than doing so in hardware. Processes upon which software random bit generators may be based include:

1. the system clock;
2. elapsed time between keystrokes or mouse movement;
3. content of input/output buffers;
4. user input; and
5. operating system values such as system load and network statistics.

The behavior of such processes can vary considerably depending on various factors, such as the computer platform. It may also be difficult to prevent an adversary from observing or manipulating these processes. For instance, if the adversary has a rough idea of when a random sequence was generated, she can guess the content of the system clock at that time with a high degree of accuracy. A well-designed software random bit generator should utilize as many good sources of randomness as are available. Using many sources guards against the possibility of a few of the sources failing, or being observed or manipulated by an adversary. Each source should be sampled, and the sampled sequences should be combined using a complex *mixing function*; one recommended technique for accomplishing this is to apply a cryptographic hash function such as SHA-1 (Algorithm 9.53) or MD5 (Algorithm 9.51) to a concatenation of the sampled sequences. The purpose of the mixing function is to distill the (true) random bits from the sampled sequences.

Menezes at p. 172 (highlighting added).

H. Rejection Sampling

79. Many cryptographic applications need random numbers that fall within a smaller range than the broad range of an underlying random number generator, which typically will generate random numbers of a certain bit-length, like 160 bits, i.e., numbers from 0 to 2^{160}-1. To use a simple example to illustrate this, suppose four friends want to decide which of them will be the designated driver for a night on the town, and they agree to decide this using a typical six-sided die, having sides showing dots reflecting the numbers 1 thru 6, respectively. The friends count off

from 1 to 4 and one of them throws the die. If the result of the throw is a number in the range 1 through 4, then it is clear who the designated driver is. But what happens when the result of the throw is ***outside*** this range? For example, suppose one them says, "Let's use modular arithmetic, so a 5 means 1 and a 6 means 2"? This is clearly a ***biased*** choice, since a friend numbered 1 or 2 will be twice as likely to be chosen as the designated driver as a friend numbered 3 or 4.

80. This bias from using a modular reduction for mapping an output from a random number generator (e.g., using a hash function) to a smaller range would be well-known to a POSA as of December 27, 2000. For example, if we desire random numbers in the range from 0 to 150, we could first generate an 8-bit random number, r (which would be in the range from 0 to 255), and then return the value r mod 151. As with the simple example given above, there is a well-known problem with this approach, however. Namely, performing such a reduction modulo q in order to get a random number in the range from 0 to q-1 will be biased towards smaller values. For instance, in the above example, in a manner analogous to the four-friends example, the number 0 is twice as likely to be output as the number 150, even if the underlying random number generator is truly random and uniform in the range from 0 to 255. For example, as I show below, the textbook LEDA

describes this bias (and how to deal with it) prior to the earliest priority date of the '961 patent.

81. A well-known solution to this problem is to use an alternative technique, which a POSA would also be familiar with, called "rejection sampling." Rejection sampling is a simple method for generating random numbers according to a specified distribution, by generating a random point in a uniform distribution and rejecting the point if it doesn't belong to the specified distribution. For example, to generate a random number uniformly chosen from the interval from 1 to q-1, for some number q, one could first generate a random number between 1 and a power of 2 greater than q-1 (which is relatively easy using a hash function starting from some seed) and then reject the sample if it is not less than q. One repeats this process until one gets a sample in the desired interval. Unlike the biased choice of reducing the result modulo q, this process is guaranteed to produce a random number that is uniformly chosen from the interval 1 to q-1. For instance, in the simple four-friends example given above, the obvious way to avoid the bias of modular arithmetic is use the rejection sampling technique, rejecting any throw of the die that results in a 5 and 6, and throwing the die again.

82. Rejection sampling is a topic that dates back to a classic 1951 paper, "Various Techniques Used in Connection with Random Digits" (**Ex. 1034**) by one

of the inventors of the digital computer, John von Neumann. According to a search I conducted Google Scholar using the keywords "Von Neumann" and "Various Techniques," this paper has been cited at least 1900 times. It is a paper that applied mathematicians and computer scientists regularly discuss and quote in our classes. Dr. von Neumann described his teachings on rejection sampling by starting with a simple example of how to get unbiased random bits by flipping a biased coin (e.g., a coin that comes up heads much more frequently than tails):

> If independence of successive tosses is assumed, we can reconstruct a 50-50 chance out of even a badly biased coin by tossing twice. If we get heads-heads or tails-tails, we reject the tosses and try again. If we get heads-tails (or tails-heads), we accept the result as heads (or tails). The resulting process is rigorously unbiased….

J. von Neumann, "Various Techniques Used in Conjunction with Random Digits," J. Res. Nat. Bur. Stand. Appl. Math., 12 (1951) (Ex. 1034), at p. 1.

83. Dr. von Neumann then explained how to generalize this approach to generating samples drawn from any distribution (e.g., a uniform distribution in the range [0,q-1] from a uniform distribution on $[0,2^{160}]$, where $q<2^{160}$) as follows:

> **One trick is to pick a scaling factor a such that $af(x)\leq 1$, and then to produce two random numbers, X and Y, from a uniform distribution on (0, 1). If $Y>af(X)$, we reject the pair and call for a new pair. If $Y\leq af(X)$, we accept X.**

Id. at p. 2.

84. For example, after years of teaching this topic, I included a discussion and homework problem on this well-known topic in my textbook, *Algorithm Design and Applications*, as follows:

> In the analysis given earlier for the Fisher-Yates shuffling algorithm to generate a random permutation, we assumed that the random(k) method, which returns a random integer in the range $[0, k-1]$, always runs in $O(1)$ time. If we are using a random-number generator based on the use of unbiased random bits, however, this is not a realistic assumption.
>
> In a system based on using random bits, we would most naturally implement the random(k) method by generating $\lceil \log k \rceil$ random bits, interpreting these bits as an unsigned integer, and then repeating this operation until we got an integer in the range $[0, k-1]$. Thus, counting each iteration in such an algorithm as a "step," we could conservatively say that the random(k) method runs in 1 step with probability $1/2$, 2 steps with probability $1/2^2$, and, in general, in i steps with probability $1/2^i$.

M.T. Goodrich and R. Tamassia, *Algorithm Design and Analysis,* Wiley, 2015(Ex. 1035), at 556 (highlighting added).

> **C-19.1** Suppose that Bob wants a constant-time method for implementing the random(k) method, which returns a random integer in the range $[0, k-1]$. Bob has a source of unbiased bits, so to implement random(k), he samples $\lceil \log k \rceil$ of these bits, interprets them as an unsigned integer, K, and returns the value $K \bmod k$. Show that Bob's algorithm does not return every integer in the range $[0, k-1]$ with equal probability.

Id. at 564 (highlighting added).

85. Rejection sampling is also a topic that is discussed in the prior art for generating random numbers in a given range using a hash function, including prior art references Miyaji and LEDA, as I discuss in detail below. In addition, two of the inventors of the '961 Patent discuss rejection sampling for generating random

numbers in a given range three years before the earliest priority date of the '961 patent in their publication, *Handbook of Applied Cryptography,* by Alfred J. Menezes, Paul C. van Oorschot, and Scott A. Vanstone, CRC Press, 1997 ("Menezes") (**Ex. 1011**). According to a search I conducted Google Scholar using the keywords "Menezes" and "Cryptography," Menezes has been cited over 20,400 times. Menezes is not cited on the face of the '961 Patent, however.

86. Menezes is a well-known handbook on applied cryptography, which is frequently used as a textbook in classes on cryptography. It includes chapters discussing random number generators, symmetric encryption/decryption, public-key cryptography, hash functions, authentication, digital signatures, key establishment protocols, and key management. For example, Chapter 5 in Menezes discloses several methods for random bit generation, including hashing a seed value using the SHA-1 cryptographic hash function, which I discuss above in ¶ 74. *See, e.g.,* Menezes at pp. 174-175. Further, on page 170, Menezes discloses a definition for random bit generation and discloses a rejection sampling method for generating uniformly random numbers in a given range from 0 to n:

5.1.1 Background and Classification

5.1 Definition A *random bit generator* is a device or algorithm which outputs a sequence of statistically independent and unbiased binary digits.

5.2 Remark (*random bits vs. random numbers*) A random bit generator can be used to generate (uniformly distributed) random numbers. For example, a random integer in the interval $[0, n]$ can be obtained by generating a random bit sequence of length $\lfloor \lg n \rfloor + 1$, and converting it to an integer; if the resulting integer exceeds n, one option is to discard it and generate a new random bit sequence.

§5.2 outlines some physical sources of random bits that are used in practice. Ideally, secrets required in cryptographic algorithms and protocols should be generated with a (true) random bit generator. However, the generation of random bits is an inefficient procedure in most practical environments. Moreover, it may be impractical to securely store and transmit a large number of random bits if these are required in applications such as the one-time pad (§6.1.1). In such situations, the problem can be ameliorated by substituting a random bit generator with a pseudorandom bit generator.

Menezes at p. 170.

87. In addition, Chapter 5 of Menezes cites the classic von Neumann paper from 1951 that I discuss above. Menezes at p. 188.

88. As I explain below, the technique of using a hash function on a seed taken from a random-number generator in conjunction with rejection sampling is disclosed in the prior art.

VII. PRIOR ART REFERENCES

89. In this section, I review prior art references that I rely on for my opinions. None of these prior art references were considered by the PTO when the application for the '961 patent was being prosecuted. And none of this prior art was considered in the earlier IPR *Facebook, Inc., et al. v. BlackBerry Ltd.*, IPR2019-00923, which I discuss below.

A. U.S. Patent 6,697,946 ("Miyaji") (Ex. 1005)

90. Miyaji is U.S. Patent 6,697,946, with a PCT filed January 28, 1997, PCT Publication date of July 30, 1998, and § 371 (c)(1), (2), (4) date of July 15, 1999. Miyaji issued on February 24, 2004. Miyaji's application was filed by Panasonic Corporation and assigned to Matsushita Electric Industrial Co. in 1999. I understand that Miyaji is prior art to the '961 Patent, since Miyaji was filed and published before the earliest priority date for the '961 Patent.

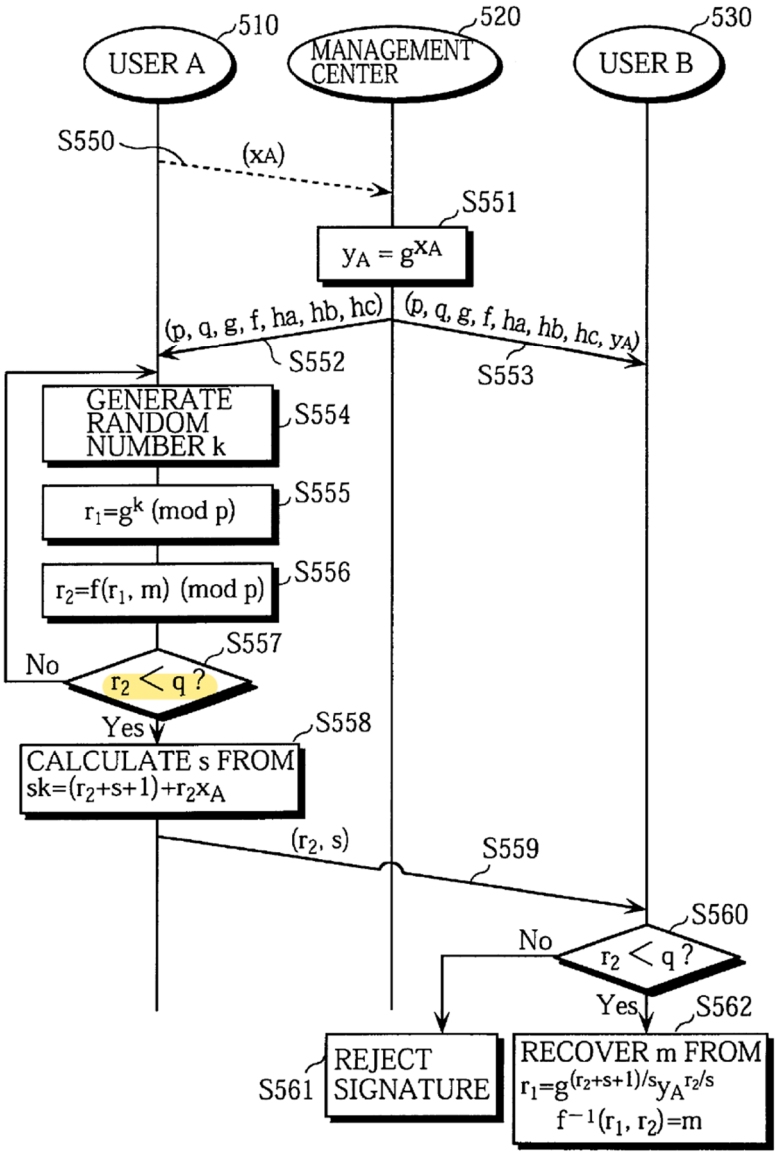
91. Miyaji discloses several cryptographic functions involving public-key cryptography, including the generation of public keys and private keys, encrypting/decrypting messages, and signing messages. *See, e.g.,* Miyaji at 1:6-10 ("The present invention relates to an apparatus for performing secret communications and digital signatures by a public key cryptosystem, and especially relates to a message recovery signature apparatus whose security is based on the discrete logarithm problem."). For example, Miyaji discloses a digital signature scheme that uses random numbers and the difficulty of the discrete logarithm problem to overcome security vulnerabilities of previous digital signature schemes. *See, e.g.,* Miyaji at 4:14-34.

92. Specifically, Miyaji discloses that a management center **520** determines a (long-term) public key y_A of a user A **510** using the user A's (long-term) private

key x_A and announces the public key y_A to a user B **530**. Miyaji, Abstract; 7:58-8:7; FIG 5. The user A **510** repeats the generation of a random number, k, which serves as an ephemeral private key, and the calculation of $r_1=g^k \pmod p$ and $r_2=f(r_1,m)=r_1+m \pmod p$ until r_2, which serves as an ephemeral public key, meets a condition, $r_2<q$, where q is the order of a generator of GF(p). *Id.* Abstract; 8:24-45; FIG. 5. If the condition is met, the user A **510** finds s to satisfy the equation $sk=(r_2+S+1)+r_2x_A \pmod q$ and sends a ciphertext (r_2,s), comprising the ephemeral public key r_2 and digital signature s, to the user B **530**. *Id.* Abstract; 8:47-56; FIG. 5. The user B **530** rejects the ciphertext if $q \leqq r_2$. *Id.* Abstract; 8:43-45; FIG. 5. If $r_2<q$, the user B **530** recovers a message m by calculating $r_1=g^k=g^{(r2+s+1)/s}y_A{}^{r2/s} \pmod p$ and $f^{-1}(r_1,r_2)=m \pmod p$, which serve as the decryption steps to verify the digital signature s using the ephemeral public key r_2. *Id.* Abstract; 8:47-54; FIG. 5. Through this procedure, a highly secure message recovery signature apparatus is realized. *Id.* Abstract.

93. As I explain in more detail below, in my opinion, Miyaji discloses the same random number generation, hashing, and rejection sampling methods of the '961 patent, e.g., as is illustrated in Fig. 5 from Miyaji, excerpted below, which provides a flowchart description of the method I summarized in the previous paragraph:

FIG. 5



Miyaji at Fig. 5 (highlighting added).

94. The step 557 in which a calculation is made whether $r_2<q$ is precisely the same step as reflected in Figure 2 of the '961 patent where it is determined whether or not H(seed) is less than q. *Compare* Miyaji, FIG. 5 *with* '961 patent, Fig. 2 & 4:11-17.

B. Bellare (Ex. 1006)

95. Bellare is "'Pseudo-random' Number Generation Within Cryptographic Algorithms: The DDS case," by Mihir Bellare, Shafi Goldwasser, and Daniele Micciancio, which was published in Proceedings of the Annual International Cryptology Conference (CRYPTO), pp. 277-291, Springer, 1997, as a part of the *Lecture Notes in Computer Science* (*LNCS*) book series, volume 1294. I understand that libraries subscribe to the LNCS book series, and I understand that Bellare is prior art to the '961 Patent, as it was publicly available and indexed in a manner that could be found by a POSA before the earliest priority date for the '961 Patent. I ran a Google Scholar search on the article co-authored by Bellare, and the information returned reflects that Bellare has been cited at least 115 times.

96. Bellare shows that if the random numbers generated in the Digital Signature Standard (DSS) Digital Signature Algorithm (DSA) are generated according to a linear congruential random number generator, then signatures generated by the algorithm are vulnerable to an attack that can reveal the secret key used. Bellare states: "This illustrates the high vulnerability of the DSS to weaknesses in the underlying random number generation process."

97. Bellare describes "the scheme" of the DSA algorithm as follows:

THE SCHEME. The scheme uses the following parameters: a prime number p, a prime number q which divides $p-1$ and an element $g \in Z_p^*$ of order q. (Chosen as $g = h^{(p-1)/q}$ where h is a generator of the cyclic group Z_p^*). These parameters may be common to all users of the signature scheme and we will consider them as fixed in the rest of the paper. The standard asks that $2^{159} < q < 2^{160}$ and $p > 2^{511}$. We let $G = \{ g^\alpha : \alpha \in Z_q \}$ denote the subgroup generated by g. Note it has prime order, and that the exponents are from a field, namely Z_q.

The secret key of a user is a random integer x in the range $\{0, \ldots, q-1\}$, and the corresponding public key is $y = g^x \bmod p$. DSA (the Digital Signature Algorithm that underlies the standard) can be used to sign any message $m \in Z_q$, as follows. The signer generates a random number $k \in \{1, \ldots, q-1\}$, which we call the *nonce*. It then computes the values $\lambda = g^k \bmod p$ and $r = \lambda \bmod q$. It sets $s = (xr + m) \cdot k^{-1} \bmod q$, where k^{-1} is the multiplicative inverse of k in the group Z_q^*. The signature of message m is the pair (r, s) and will be denoted by $\mathrm{DSA}(x, k, m)$. Note that a new, random nonce is chosen for each signature.

A purported signature (r, s) of message m can be verified, given the user's public key y, by computing the values $u_1 = m \cdot s^{-1} \bmod q$, $u_2 = r \cdot s^{-1} \bmod q$ and checking that $(g^{u_1} y^{u_2} \bmod p) \bmod q = r$. Notice that the values (r, s) output by $\mathrm{DSA}(x, k, m)$ satisfy the relation $sk - rx = m \pmod q$. We will make use of this relation in our attack on the DSS.

Bellare at 281.

98. That is, a POSA as of December 27, 2000 would understand that Bellare describes "the scheme" of DSA. This scheme begins by choosing a large prime number $p>2^{511}$, represented using bit string of 512 bits, and a prime number $q>2^{159}$, represented using a bit string of 160 bits, such that q is the order of a generator, g, of Z^*_p. The schemes uses a private key, x, which is a random integer in the range {0, …, q-1}, and a corresponding public key, $y=g^x \bmod p$; hence, the security of y is based on the Discrete Logarithm problem. The scheme of DSA then

generates a random number k in the range from 1 to q-1, which it then uses as an ephemeral private key for producing the signature, s, of a message, m, according to the steps outlined above.

99. Bellare discloses that, in addition to the security of the long-term public key, y, the security of DSA depends critically on the secrecy of the random number k (called a "nonce") that is used:

> **SECRECY OF THE NONCE. Recall that for every signature, the signer generates a new, random nonce k. An important feature (drawback!) of the DSS is that the security relies on the secrecy of the nonces. If any nonce k ever becomes revealed, at any time, even long after the signature (r, s) was generated, then given the nonce and the signature one can immediately recover the secret key x, via $x = (sk - m)r^{-1} \bmod q$. This is a key point in our attack.**

Bellare at 281.

C. LEDA (Es. 1007)

100. LEDA is *LEDA: a Platform for Combinatorial and Geometric Computing*, by Kurt Mehlhorn and Stefan Näher, Cambridge University Press, 1999. I understand that LEDA is prior art to the '961 patent, as it was publicly available and indexed in a manner that could be found by a POSA before the earliest priority date for the '961 Patent. In fact, according to a search of the article I ran on Google Scholar, LEDA has been cited over 1,400 times.

101. LEDA is a well-known textbook on methods for implementing data structures and algorithms in the programming language C++. It includes disclosures

on basic data types (including random sources), hash functions, numbers and matrices, search trees, graphs, geometric data types, convex hulls, triangulations, point sets, sorting, network algorithms, and planarity testing.

102. As I explain in more detail below, in my opinion, LEDA discloses the rejection sampling random-number generation algorithm of the '961 Patent:

> **Implementation of random_source:** Our implementation of random sources follows the description in [Knu81, Vol2, section 3.2.2]. We first give the mathematics and then the program. Internally, we always generate a sequence of integers in the range $[0\,..\,2^{31}-1]$. We define 32 unsigned longs $X_0, X_1, \ldots, X_{31}$ by
>
> $$X_0 = seed$$
>
> and
>
> $$X_i = (1103515245 \cdot X_{i-1} + 12345) \bmod m$$
>
> for $1 \leq i \leq 31$. Here $m = 2^{32}$. We extend this sequence by
>
> $$X_i = (X_{i-3} + X_{i-32}) \bmod m$$
>
> for $i \geq 32$. In this way an infinite sequence $X_0, X_1, \ldots$ of unsigned longs is obtained. Following [Knu81, Vol2, section 3.2.2], we discard the first 320 elements of this sequence (they are considered as a warm-up phase of the generator) and we also drop the right-most bit of each number (since it is the least random). Thus, the i-th number output by the internal generator is
>
> ```
> (X[i + 320] >> 1) & 0x7fffffff.
> ```
>
> We next show how to generate a number uniformly at random in $[low\,..\,high]$. Let X be a number produced by the internal generator. Then $low+X \bmod(high-low+1)$ is a number in the range $[low\,..\,high]$. However, this number is not uniformly distributed (consider the case where $low = 0$ and $high = 2^{31}-2$ and observe that in this case the number 0 is generated with probability twice as large as any other number). We therefore proceed differently. Our

LEDA at p. 92.

approach is based on the observation that if X is a random number in $[0 .. 2^{31}-1]$ and p is an integer less than 32 then $X \bmod 2^p$ is a random number in $[0 .. 2^p-1]$. Let $diff = high-low$ and let p be such that $2^{p-1} \leq diff < 2^p$. We generate random numbers X using the internal source until $X \bmod 2^p \leq diff$ and then output $low + X \bmod 2^p$. Since $2^{p-1} \leq diff$ at most two X's have to be tried on average. The complete program follows.

⟨*generation of a random number in [low..high]*⟩≡

```
int diff = high - low;
/* compute pat = 2^p - 1 with 2^{p-1} <= diff < 2^p */
unsigned long pat = 1;
while (pat <= diff) pat <<= 1;
pat--;
/* pat = 0...01...1 with exactly p ones.
   Now, generate random x in [0 .. pat]
   until x <= diff and return low + x    */
unsigned long x = internal_source() & pat;
while ( x > diff) x = internal_source() & pat;
return (int)(low + x);
```

LEDA at p. 93.

103. A POSA would understand that the equations on page 92 disclose the use of a random number generator, using a hash function, which is initialized with a random seed. Starting with the first number generated, each number returned by the "random source," which is denoted "internal_source" on page 93, is generated from a hash function applied to a seed value from a random number generator. The step that performs the rejection sampling and repetition is the following:

```
while ( x > diff) x = internal_source() & pat;
```

LEDA at 93.

104. A POSA would recognize that the "& pat" portion in the above code is an "and" function (which is not a mod function) that performs a bitwise-and with the mask "pat" so that the result is a hash value in the range 0 to 2^p-1, that is, a p-bit number.

D. Facebook's Prior Art

105. I understand that Facebook and two other related entities previously filed a Petition for Inter Partes Review of the '961 patent, relying upon different prior art than that which is raised in the present Petition: *Facebook, Inc., et al. v. BlackBerry Ltd.*, 2019IPR00923 (PTAB) (**Ex. 1018**). In particular, I understand that Facebook raised the following four obviousness challenges: (a) claims 1, 2, 5, 15, 16 and 19 were obvious based on the combination of ("DSS") (using DSS 186) (**Ex. 1004**), in view of Schneier, Applied Cryptography (2d ed. 1996) (**Ex. 1020**) ("Schneier") and Rose, Re: "Card-shuffling" algorithms, USENET, sci.crypt, sciath h 10, 1993) (**Ex. 1021**) ("Rose"); (b) claims 1, 2, 5, 15, 16 and 19 were obvious based on the combination of DSS, Schneier, and Menezes (**Ex. 1011**)[16]; (c) claims 23, 24 and 27 were obvious based on the combination of DSS, Schneier, Rose, Rao,

[16] In addition to the pages from the Menezes treatise that were offered as an Exhibit and referenced by Facebook in its IPR Petition, I make reference in this Declaration to other pages from that same treatise. I understand that the additional pages I cite have been included in Exhibit 1011.

Error Coding for Arithmetic Processors (1974) (**Ex. 1022**) ("Rao") and Floyd, Essentials of Data Processing (1987) (**Ex. 1023**) ("Floyd"); and (d) claims 23, 24 and 27 were obvious based on the combination of DSS, Schneier, Menezes, Rao and Floyd.

106. I have reviewed several of the filings from the Facebook IPR, and I am aware that the Board denied institution of Facebook's Petition in a Decision issued on November 5, 2019. *Id.*, Paper 14 (**Ex. 1019**). In particular, I understand that the Board found that none of the combinations taught the required step in each of independent claims 1, 15 and 23 that required evaluating the output of a hash function performed on a random number to determine if that output is less than a group of order q, and, if it is not, rejecting that value and repeating the method – what the Board called the "determining," "accepting," "rejecting," and "repeating" steps of the independent claims. *See id.* at 13-15. For instance, the Board found that DSS simply performs a modulo operation on the output of the hash of a random value and stores that result as ephemeral private key k, and does not contemplate evaluating the output of the hash function prior to performing the modulo operation, much less iteratively repeating the process of selecting a new random number and performing a hash function on that number until a value less than q is obtained. *Id.* at 14. Similarly, the Board concluded that Rose does not disclose hashing the

random number at any point in its method, or contemplate how its technique might be applied to a cryptographic key generation algorithm, whether its technique might be useful in avoiding bias in the output of a hash function performed on a random number, or whether avoidance of modulo bias requires iteratively repeating both the selection of a random number and hashing that number. *Id.* at 14-15.

107. For similar reasons, I understand that the Board also concluded that the portions of Menezes relied upon by Petitioner fail to disclose the "determining," "accepting," "rejecting," and "repeating" steps performed on the output of a hash function as required by the challenged claims. *Id.* at 19. Instead, the Board found that Menezes does not mention hashing, and does not contemplate, for example, whether its technique might be useful for accepting or rejecting the result of performing a hash function on a randomly generated seed value, or whether iterative repetition of both the generation of a random number and performance of a hash function on that number when the output of the hash function falls outside of a desired range would be appropriate. *Id.* at 20.

108. In my opinion, the prior art that Petitioner raises in this Petition, which was not before the Examiner during prosecution of the '961 patent, easily overcomes the issues identified by the Board in the *Facebook* IPR. Miyaji, as well as Bellare and LEDA, which I analyze below with respect to anticipation and

obviousness of the challenged claims, explicitly teach the "determining," "accepting," "rejecting," and "repeating" steps performed on the output of a hash function as required by the challenged claims. ***Indeed, Miyaji teaches the same algorithm as is claimed in each of the independent claims of the '961 patent.***

109. In my opinion, therefore, the prior art cited in this Declaration easily overcomes the issues that the Board identified with respect to why the prior art raised in Facebook's earlier IPR did not render the challenged claims obvious (let alone anticipated).

VIII. THE '961 PATENT

A. Summary of the '961 Patent

110. U.S. Patent 7,372,961 ("the '961 patent") was filed December 26, 2001, published as US 2002/0090085 on July 11, 2002, and issued May 13, 2008. It has a related foreign application with a priority date of December 27, 2000. As mentioned above, I understand that the earliest possible priority date for the '961 patent is December 27, 2000.

111. The '961 patent generally purports to describe a method for avoiding potential bias in the generation of a cryptographic key. In particular, the "Background of the Invention" section of the '961 patent discloses several topics as "well known," "ubiquitous," and/or "popular," which I find to be

acknowledgements in the specification of the '961 patent regarding the knowledge of a POSA. This admitted knowledge of a POSA in the "Background of the Invention" section of the '961 Patent includes the following:

- The basic structure of public-key cryptographic systems, including that a random number, k, in a given range may be used to generate a public key. '961 patent at 1:10-16, 2:15-22.
- Public-key cryptosystems may be used for key agreement, e.g., by exponentiating with a private key. *Id.* at 1:17-19.
- Public-key cryptosystems may be used for digital signatures. *Id.* at 1:26-27.
- Public-key cryptosystems may rely on the intractability of the discrete logarithm problem in finite field arithmetic. *Id.* at 1:33-37.
- The Digital Signature Algorithm (DSA), including the mathematics behind it, such as for generating the parameters p, q, and g (e.g., with p and q being prime numbers, as is guaranteed by Cauchy's theorem). *Id.* at 1:51-2:14.
- There is a modular bias in the way the (NIST) FIPS 186 (**Ex. 1004**) discloses to compute the key k for DSA. *Id.* at 2:32-55.

112. More specifically, the patent purports to remedy a modulo bias in the generation of a key *k* in connection with the Digital Signature Algorithm (DSA), described via the **Digital Signature Standard (DSS)** which I described above. DSS was published by the National Institute of Standards and Technology (NIST), which

is part of the United States Department of Commerce. *See* Federal Information Processing (FIPS) Publication 186, *Digital Signature Standard (DSS)* (May 19, 1994), (**Ex. 1004**), p.1. The signature algorithm is commonly known as the DSA, and the published document that describes the standard is known as the DSS.

113. The DSS explains that "[a] digital signature is an electronic analogue of a written signature in that the digital signature can be used in proving to the recipient or a third party that the message was, in fact, signed by the originator." DSS FIPS 186 (**Ex. 1004**), p.5, § 2. "Digital signatures may also be generated for stored data and programs so that the integrity of the data and programs may be verified at any later time." *Id.*

114. As my discussion of the standards above reflects, neither DSA nor DSS were inventions of the '961 patent. By the time the earliest application for the '961 patent was filed in December 2000, NIST had already published three versions of DSS – in 1994 (FIPS 186) (**Ex. 1004**), 1998 (FIPS 186-1) (**Ex. 1032**), and 2000 (FIPS 186-2) (**Ex. 1033**). The Background section of the '961 patent specification specifically cites and discusses FIPS 186-2, the 2000 version of DSS. '961 patent, 2:36-46. For purposes of my analysis, I will cite primarily to the first version of DSS (FIPS 186/1994) (**Ex. 1004**), because it is the oldest version of the DSA/DSS.

The description of how to generate keys in DSS, as summarized in the '961 patent, did not materially change between FIPS 186 (1994) and FIPS 186-2 (2000).

115. The '961 patent focuses on one narrow aspect of DSS – the generation of the ephemeral cryptographic key, *k*. As summarized by the '961 patent, the DSS specifies a method for generation of key *k* using a random number generator and a hashing algorithm, "G( )," and requires that the key *k* be less than "q", a large prime number. '961 patent, 1:56-64, 2:40-43; DSS, **Ex. 1004**, pp. 6, 13-15.

116. As in LEDA (**Ex. 1007**) and the above-referenced homework question in my textbook, *Algorithm Design and Applications* (**Ex. 1035**), the '961 patent observes the need for a random number, k, which is used in the DSA and ECDSA algorithms, to be uniformly distributed in the range from 1 to q-1, for a parameter, q. As explained in the '961 patent's specification:

> Neither the DSA nor the ECDSA inherently disclose any information about the public key k. They both require the selection of k to be performed by a random number generator and it will therefore have a uniform distribution throughout the defined interval. However the implementation of the DSA may be done in such a way as to inadvertently introduce a bias in to the selection of k. This small bias may be exploited to extract a value of the private key d and thereafter render the security of the system vulnerable. One such implementation [of the DSA] is the DSS mandated by the National Institute of Standards and Technology (NIST) FIPS 186-2 Standard. The DSS stipulates the manner in which an integer is to be selected for use as a private key. A seed value, SV, is generated from a random number generator which is then hashed by a SHA-1 hash

> function to yield a bit string of predetermined length, typically 160 bits. The bit string represents an integer between 0 and 2^{160}-1. However ***this integer could be greater than the prime q and so the DSS requires the reduction of the integer mod q, i.e. k=SHA-1(seed) mod q.***

'961 patent, 2:28-46 (emphasis added).

117. To summarize, the DSS first specifies generating a random number seed, and then computing a hash of the random number SHA-1(seed). Because the value of SHA-1(seed) may be larger than the desired range (as determined by the "prime q"), a "mod q" operation is performed to compute the value of the cryptographic key k, *i.e.,* k=SHA-1(seed) mod q. '961 patent, 2:44-46. Computing "SHA-1(seed) mod q," as explained above, ensures that the value of the key k will fall within the desired range required by the DSA, *i.e.,* from 0 to q-1.

118. But as also explained above, this method for generating *k* could introduce some modulo bias in favor of values on the smaller end of the range. The '961 patent acknowledges that the possibility of such bias was known. '961 patent, 2:53- 55 ("With this algorithm it is to be expected that more values will lie in the first interval than the second and therefore there is a potential bias in the selection of k.").

119. The apparent impetus behind the patent was work by a third party, Dr. Daniel Bleichenbacher, who attempted to quantify the bias and suggested that it

might reduce the effort needed to recover a private key in DSA. The patent explains that "[r]ecent work by Daniel Bleichenbacher," a known cryptographer who is not named as an inventor on the '961 patent, "suggests that the modular reduction to obtain k introduces sufficient bias in to the selection of k that an examination of 2^{22} signatures could yield the private key d in 2^{64} steps using 2^{40} memory units." '961 patent, 2:56-59. "This suggests that there is a need for the careful selection of the ephemeral key k." *Id.*, 2:60-61.

120. The '961 patent purports to solve the problem identified by Bleichenbacher in the same manner as the prior art – instead of relying on a modular reduction to obtain values within the desired range (from 0 to q-1), the '961 patent simply *rejects* any random numbers that are not less than q, and keeps generating new random numbers until one is obtained that is less than q. *Id.*, 4:11-17.

121. The '961 patent addresses this apparent potential bias in the random generation of a key, k, by using the well-known rejection sampling method, where one first inputs a seed output from a random number generator as an input to a hash function and accepts the result if it is less than a parameter, q. If the result is rejected, because the hashed seed is not less than q, then the method repeats. This simple process is shown, for example, in Fig. 2 in the '961 patent:

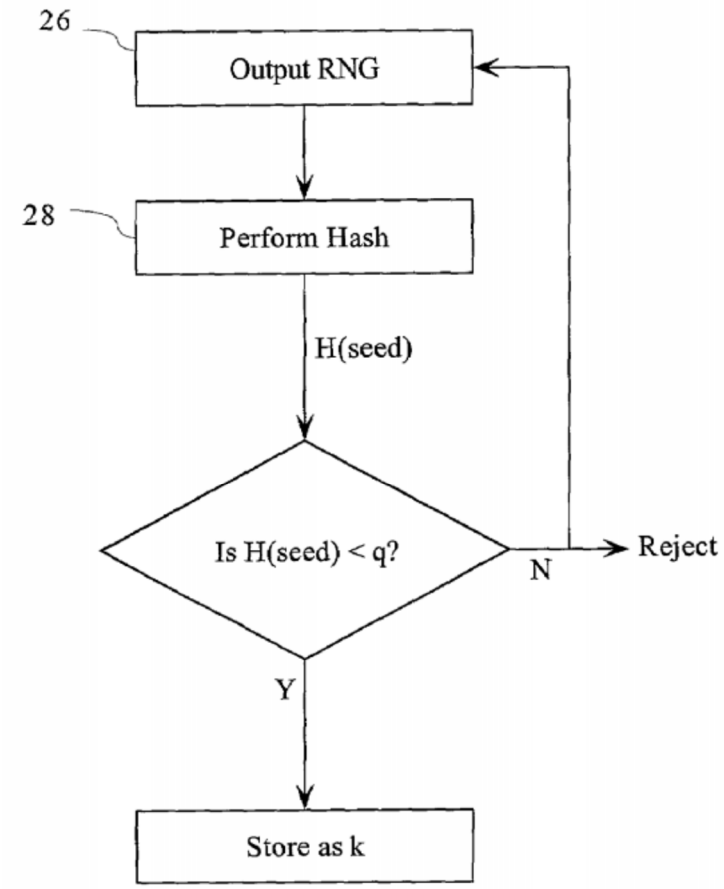


Fig. 2

'961 Patent, Fig. 2.

Figure 2 provides "a flow chart showing a first embodiment of key generation" (*id.*, 3:12-13), and that embodiment is the one reflected in the challenged claims. The method "originates by obtaining a seed value (SV) from the random number generator 26." *Id.*, 3:64-66. The number output from the random number generator (RNG) is then provided to hash function 28 in order to produce an ouput – H(seed)

of the correct length. *Id.* 4:6-10. This is the same value represented as "SHA-1(seed)" in the Background description of DSA. *Id.*, 2:40-46. 17. The '961 patent explains that the hashing function is used "to yield a bit string of predetermined length, typically 160 bits." *Id.*, 2:40-43, 3:38-41. The remaining steps in Figure 2 perform the simplified rejection technique of the prior art – namely, they check if H(seed) is less than q, where the maximum desired value is q-1. If the answer is "yes," H(seed) is accepted and used as the cryptographic key k. Otherwise the value H(seed) "is rejected and the random number generator is conditioned to generate a new value which is again hashed by the function 28 and tested. This loop continues until a satisfactory value is obtained." *Id.*, 4:13-17.

B. The File History for the '961 Patent

122. The '961 patent was filed on December 26, 2001 as U.S. Application 10/025924. The initial application had 14 claims, 2 of which were independent claims. **Ex. 1003**, at 1. The file history reflects repeated rejections of the claims as being anticipated or rendered obvious by the prior art, and the Applicant repeatedly amending the claims to address such art. Notably, in my opinion none of the many prior art references cited by the Examiner as being relevant to the claims are as relevant to the subject matter as the references I cited and discuss in this Declaration.

Also of note is that none of the references cited on the face of the '961 patent were supplied by the Applicant—they all were provided by the Examiner.

123. Initially, all 14 claims as originally drafted by the Applicant were rejected in a non-final office action mailed March 2005, as being anticipated by Schneier, *Applied Cryptography* (pages 483-490, which describe DSA), and/or being rendered obvious over Schneier in view of Backal (U.S. Patent 6,219,421). **Ex. 1003**, at 66-76.

124. The applicant then made amendments to both the specification and the claims on September 22, 2005, and argued that Schneier and Backal did not teach the rejection and repetition steps of the claims. *Id.* at 81-95. The amended claims were then further rejected in a December 6, 2005 office action, as being obvious over the combination of Schneier in view of Matyas (U.S. Patent 6,307,938), which discloses using a SHA-1 hash on a seed to produce hash values until it satisfies a primality test, Backal, and Nel ("Generation of Keys for use with Digital Signature Standard (DSS)"). *Id.* at 110-21.

125. The Applicant responded on March 3, 2006, by once again amending the claims, e.g., so that the order q is limited to be prime in dependent claim 5 instead of independent claim 1.

1. (currently amended) A method of generating a key over a group of order q, said method including the steps of:

generating a seed value from a random number generator;

performing a hash function on said seed value to provide an output;

determining whether said output is less than said ~~prime number~~ order q;

accepting said output for use as a key if the value thereof is less than said ~~prime number~~ order q; and

rejecting said output as a key if said value is not less than said order q.

5. (currently amended) The method of claim 1 wherein said order q is a prime number represented by a bit string of predetermined length L.

Id. at 126.

126. Applicant also argued that Matyas didn't teach using a hash function as in the proposed claims and, e.g., comparing the hash value to q to avoid bias. *Id.* at 129-33.

127. The amended claims were then again rejected in an office action dated April 13, 2006, as being obvious over the previously cited art, such as the use of a one-way hash function with a seed. *Id.* at 138-51.

The applicant argued Matyas does not teach, "performing a hash". The Examiner disagrees. Matyas teaches generating a seed value and hashing each of these seed values with SHA-1 to produce corresponding hash values. Therefore, Matyas teaches generating a seed and generating hash value by performing a hash function on the seed value. (Col. 5, line 65- Col. 6, line 8). In addition, Matyas further disclose the seed values are also kept secret. (Col. 6, line 24)

Id. at 138.

128. The Applicant held a telephone interview with the Examiner on June 16, 2006, after which the Applicant filed a response on June 28, 2006 that amended the claims to add "cryptographic function" and "H(SV)" limitations, e.g., as follows:

1. (currently amended) A method of generating a key *k* for use in a cryptographic function performed over a group of order q, said method including the steps of:

generating a seed value *SV* from a random number generator;

performing a hash function *H()* on said seed value *SV* to provide an output *H(SV)*;

determining whether said output *H(SV)* is less than said order q;

accepting said output *H(SV)* for use as [[a]] said key *k* if the value ~~thereof~~ of said output *H(SV)* is less than said order q; [[and]]

rejecting said output *H(SV)* as [[a]] said key if said value is not less than said order q[[.]];

if said output *H(SV)* is rejected, repeating said method; and

if said output *H(SV)* is accepted, providing said key *k* for use in performing said cryptographic function, wherein said key *k* is equal to said output *H(SV)*.

Id. at 163.

129. These further amended claims were yet again rejected in an office action on November 3, 2006, as unpatentable over Vanstone (U.S. Patent 6,195,433) in view of Schneier and Matyas, as well as Backal and Nel, and further in view of Patel (U.S. Patent 6,327,660). *Id.* at 182-98.

130. The Applicant responded by arguing against this rejection, and the claims were rejected in a subsequent office action on March 20, 2007. The Applicant responded, after an interview with the Examiner, by amending the claims to add the limitation "prior to reducing mod q" to claim 1, keeping claims 2-8 unchanged:

> 1. (currently amended) A method of generating a key *k* for use in a cryptographic function performed over a group of order q, said method including the steps of:
>
> generating a seed value *SV* from a random number generator;
>
> performing a hash function *H()* on said seed value *SV* to provide an output *H(SV)*;
>
> determining whether said output *H(SV)* is less than said order q prior to reducing mod q;
>
> accepting said output *H(SV)* for use as said key *k* if the value of said output *H(SV)* is less than said order q;
>
> rejecting said output *H(SV)* as said key if said value is not less than said order q;
>
> if said output *H(SV)* is rejected, repeating said method; and
>
> if said output *H(SV)* is accepted, providing said key *k* for use in performing said cryptographic function, wherein said key *k* is equal to said output *H(SV)*.

Id. at 252.

131. The Applicant also added new claims 15-30, which are parallel claims to the current (and as issued) claims 1-8 for a computer readable medium and

cryptographic unit, respectively. *Id.* at 254-56. Significantly, in making this Amendment, the Applicant explained that the DSA/DSS "requires the reduction of the integer mod q which can expect that more values will lie in the first interval than the second, hence the bias." *Id.* at 257.

132. The Examiner proposed an amendment to the cryptographic unit claims to add "an arithmetic processor" limitation, on January 18, 2008. The amended claims were accepted, and the patent issued May 13, 2008.

C. The Challenged Claims

133. My Declaration addresses claims 1-6, 15-20, and 23-28, of which claims 1, 15, and 23 are independent. Claims 1-6 are representative and reflect the first embodiment from the specification described above:

1. **[a]** A method of generating a key k for use in a cryptographic function performed over a group of order q, said method including the steps of:

[b] generating a seed value SV from a random number generator;

[c] performing a hash function H( ) on said seed value SV to provide an output H(SV);

[d] determining whether said output H(SV) is less than said order q prior to reducing mod q;

[e] accepting said output H(SV) for use as said key k if the value of said output H(SV) is less than said order q;

[f] rejecting said output H(SV) as said key if said value is not less than said order q;

[g] if said output H(SV) is rejected, repeating said method; and

[h] if said output H(SV) is accepted, providing said key k for use in performing said cryptographic function, wherein said key k is equal to said output H(SV).

2. The method of claim **1** wherein another seed value is generated by said random number generator if said output is rejected.

3. The method of claim 1 wherein the step of accepting said output as a key includes a further step of storing said key.

4. The method of claim 1 wherein said key is used for generation of a public key.

5. The method of claim **1** wherein said order q is a prime number represented by a bit string of predetermined length L.

6. The method of claim 5 wherein said output from said hash function is a bit string of predetermined length L.

'961 patent, 5:33-62. Many of the limitations in these claims recite features that are admitted prior art from the DSA/DSS, including everything in limitations 1**[a]**, 1**[b]**, and the extra feature added by dependent claims 3-5. The other two groups of challenged claims (*i.e.,* claims 15-20 and claims 23-28) recite the computer readable medium and apparatus claims corresponding to the subject matter in claims 1-6, respectively.

D. Proposed Claim Constructions for the '961 Patent

134. I have been asked to provide an opinion regarding the interpretation of one phrase recited in each independent challenged claim of the '961 patent –

"**reducing mod q**." I understand that the construction of this phrase is the subject of an ongoing dispute in the litigation between the Facebook, Inc. and the Patent Owner in ongoing patent litigation occurring in the Central District of California.

135. I understand that in the Central District litigation and in a prior *Inter Partes* petition directed to the '961 patent, Patent Owner unsuccessfully argued that: "'reducing mod q' has a plain and ordinary meaning that refers to lowering a numerical value based on a modulo q." I respectfully disagree, and instead agree with the claim interpretation that I understand Facebook has proposed, and which both the Board and the judge supervising the Central District litigation has adopted. That is, as I explain below, in the field of cryptography, the words "reducing" and "reduction" when used in conjunction with "mod q" do not require lowering of a numerical value based on a modulus q.

136. A modular reduction, expressed mathematically as "a mod b," returns the remainder when a is divided by b. It will generate a lower number when $a \geq b$, but it will not generate a lower number when $a<b$. A modular reduction therefore may not result in a lower value, depending on the inputs to the operation. Nonetheless, persons of ordinary skill in the art refer to "a mod b" as performing a "modular reduction" or a "reduction modulo b" in either circumstance, regardless of whether or not the resulting value is less than the original "a" value.

137. I will provide a simple example. Assume we are using a modulus of 3 (i.e., b=3). Then if a is 5, the result of computing "5 mod 3" is 2, because 5 divided by 3 returns 1, with a remainder of 2. Modular reduction in this case does result in a lower value. However, if a is 2, then the result of computing "2 mod 3" is 2, because 2 divided by 3 returns 0, with a remainder of 2. Both are examples of "modular reduction" based on the modulus 3, or just "reducing modulo 3."

138. From my review of the IPR and Central District litigation pleadings, Patent Owner's approach to determining the meaning of "reducing mod q" appears to construe the word "reducing" in isolation using lay-person dictionary definitions. I disagree with this approach. Construing the term "reducing mod q" (also called "modular reduction" using a modulus of q) requires interpreting the entire term, as the entire term is well-known and defined in the field of cryptography. Thus, treating the parts of the term separately does not capture the true technical meaning of the larger "reducing mod q" phrase.

139. In the context of mathematics and cryptography, the phrase "reducing mod q" simply refers to performing the mod q operation, i.e., computing the remainder when dividing a value by q. As explained extensively above, the operation expressed as "**a mod b**" *will* generate a smaller number when a ≥ b, but

will not when b > a. A modular reduction therefore may or may not result in a reduced value, depending on the values that are input into the operation.

140. The fact that "reducing mod q" simply refers to computing the remainder upon division by q is clear from the '961 patent. The specification refers to reducing mod q only once: "However this integer [SHA-1(seed)] could be greater than the prime q and so the DSS requires the **reduction** of the integer mod q, **i.e.** k=SHA−1(seed) mod q ('961, 2:44-46 (emphasis added)). This sentence treats "reduction of the integer mod q" as synonymous with computing "SHA-1 (seed) mod q." Because the integer SHA-1 seed *could* be greater than q, it must be reduced modulo q. If SHA-1 (seed) was less than q, reducing it modulo q does not lower that value, but if SHA-1(seed) was greater than q, then reducing it modulo q lowers the value.

141. Further, Menezes (*Handbook of Applied Cryptography*), which includes two of the inventors of the '961 Patent as coauthors, describes a "Reduction modulo m" algorithm that takes as input an integer, x, and a modulus, m, and outputs "r = x mod m." This algorithm is specialized for a modulus of the form, $m = b^t - c$, but a POSA would understand that its use of the phrase "reduction modulo m" is generic:

14.3.4 Reduction methods for moduli of special form

When the modulus has a special (customized) form, reduction techniques can be employed to allow more efficient computation. Suppose that the modulus m is a t-digit base b positive integer of the form $m = b^t - c$, where c is an l-digit base b positive integer (for some $l < t$). Algorithm 14.47 computes $x \bmod m$ for any positive integer x by using only shifts, additions, and single-precision multiplications of base b numbers.

14.47 Algorithm Reduction modulo $m = b^t - c$

INPUT: a base b, positive integer x, and a modulus $m = b^t - c$, where c is an l-digit base b integer for some $l < t$.
OUTPUT: $r = x \bmod m$.

1. $q_0 \leftarrow \lfloor x/b^t \rfloor$, $r_0 \leftarrow x - q_0 b^t$, $r \leftarrow r_0$, $i \leftarrow 0$.
2. While $q_i > 0$ do the following:
   2.1 $q_{i+1} \leftarrow \lfloor q_i c/b^t \rfloor$, $r_{i+1} \leftarrow q_i c - q_{i+1} b^t$.
   2.2 $i \leftarrow i + 1$, $r \leftarrow r + r_i$.
3. While $r \geq m$ do: $r \leftarrow r - m$.
4. Return(r).

Menezes at p. 605 (highlighting added).

142. That "reducing mod q" does not require lowering a numerical value also finds support in the claim language. Claim 1**[c]** reads: "determining whether said output H(SV) is less than said order q prior to reducing mod q." The purpose of this step is to ensure that any reduction of H(SV) mod q (*i.e.,* computing "H(SV) mod q") must occur ***after*** the determination of whether H(SV) is less than q. The claim goes on to reject H(SV) and repeat the method if H(SV) is not less than q, thus generating a new random number and a new H(SV). The claim, in other words, requires repeatedly generating a new SV and new H(SV) until H(SV) < q. Under the plain language of the claim, therefore, "reducing mod q" would ***never*** result in a reduction of the value of H(SV) – the operation "H(SV) mod q" would ***always*** result

in the same value because H(SV) is always less than q when H(SV) is accepted, and thus reduction mod q might occur.

143. Accordingly, the fact that the claim refers to this operation as "reducing mod q" – even though the claim specifically is designed to guarantee that this operation will ***not*** result in a reduction in the value of H(SV) – provides compelling evidence that the phrase "reducing mod q" is not limited to cases where a value actually is reduced.

144. For these reasons, in my opinion, the claim terms in the '961 Patent should be given their meanings as in IPR2019-00923 and the Central District litigation. That is, "reducing mod q" should mean "computing the remainder of dividing a value by q," and all other claim terms should be given their ordinary and customary meanings as understood by a POSA and the prosecution history pertaining to the patent. These are the claim constructions that I have used in my analysis. My opinions would not change, however, if other constructions are adopted that are similar to these constructions.

IX. ANTICIPATION

A. Miyaji

145. In my opinion, Miyaji anticipates claims 1-4, 15-18, and 23-26 of the '961 patent.

i. Claims 1[a], 15[a], and 23[a]

146. In my opinion, Miyaji anticipates claims 1, 15, and 23 of the '961 Patent. I describe how each limitation of claims 1, 15, and 23 are disclosed by Miyaji below. This includes the preambles of each claim:

> **1[a]: "A method of generating a key k for use in a cryptographic function performed over a group of order q, said method including the steps of:" 1[a]: "A method of generating a key k for use in a cryptographic function performed over a group of order q, said method including the steps of:"**
>
> **15[a]: "A computer readable medium comprising computer executable instructions for generating a key k for use in a cryptographic function performed over a group of order q, said instructions including instructions for:**
>
> **23[a]: "A cryptographic unit configured for generating a key k for use in a cryptographic function performed over a group of order q, said cryptographic unit having access to computer readable instructions and comprises:"**

147. To the degree that the preamble of claim 1 is limiting, Miyaji discloses limitations 1[a], 15[a] and 23[a], "A method of generating a key k for use in a cryptographic function performed over a group of order q, said method including the steps of."

148. For example, Miyaji discloses its field of invention as follows:

> The present invention relates to an apparatus for performing secret communications and digital signatures by a **public key cryptosystem**, and

> especially relates to a message recovery signature apparatus whose security is based on the discrete logarithm problem.

Miyaji at 1:5-10 (emphasis added).

149. A POSA would understand that the method of Miyaji involves cryptographic functions (e.g., in public-key cryptography) performed over a group of order q, and that Miyaji discloses methods to generate a number of keys, including a key, r_2. For example, a POSA would understand the digital signature method of Miyaji to involve first setting up system parameters that include a prime number, p, and an element, g, that is a member of the field GF(p), where all arithmetic is modulo p, such that q is the order of g. *See, e.g.,* Miyaji at 4:12-34 and 7:9-23 ("The message recovery signature scheme used in this system is a public key cryptosystem that uses the discrete logarithm problem as the founding principle for the security, and is based on operations over a finite field GF(p) where p (512 bits long) is a prime number, g is an element, and q (256 bits long) is the order of g."). To digitally sign a message, m, the method begins by generating a random number, k, from a random number generator, which serves as a seed. Miyaji at 7:16-21. Next, the method computes a hash of the seed, k, according to the following formula:

$$r_2 = H(k) = g^k + m \pmod{p}.$$

See, e.g, id. at 8:32-41.

150. Miyaji teaches a method of generating a key k for use in a cryptographic function, that is, generating the key, r_2, and the digital signature, s. Here, I am using the notation "H(k)" to denote the hash function taught in Miyaji that consists of composing two hash functions disclosed in Miyaji. For example, a POSA would understand H(k) to be an exponential/discrete-logarithm hash function, $g^k \pmod p$, which Miyaji stores in an intermediate variable, r_1, combined with the multiplicative hash function, $f(r_1,m) = r_1 + m \pmod p$, disclosed in Miyaji. That is, a POSA would understand Miyaji's disclosures regarding r_1 and r_2 as follows:

$$\begin{aligned} r_2 &= f(r_1, m) \\ &= r_1 + m \pmod p \\ &= g^k \pmod p + m \pmod p \\ &= g^k + m \pmod p \\ &= H(k). \end{aligned}$$

See, e.g., Miyaji at 8:23-35.

151. The process taught in Miyaji then compares r_2 to q. If r_2 is not less than q, then the method repeats the above steps, choosing a new seed, k, and generating r_2 according to the hash function, H(k), given above, with all other parameters unchanged. *Id.* at 7:23-45. If $r_2 < q$, then the method generates a signature, s, using the key r_2, so as to satisfy the following equation:

> sk = $(r_2 + s + 1) + r_2x_A \pmod q$, where x_A is a long-term secret key paired with a long-term public key, y_A, for user A.

Id. at 7:56-8:8, 8:47-53.

152. That is, r_2 is repeatedly generated, compared to q until it is less than q, and then accepted, all prior to a reduction modulo q.

153. Miyaji describes r_2 as a "masked message" that is generated from the "commitment" r_1 (which is an exponentiation from a random private nonce, k) and the message "m." Miyaji at 4:23-26, 5:5-6; claim 1. Miyaji also identifies r_2 as being part of the "ciphertext (r_2,s)" that is sent to the user B for decryption. *Id.* Abstract, 2:12-13, 2:31-32.

154. As Miyaji's generation of r_2 reflects, a POSA would understand r_2 to be a key[17] for the signature, s, because r_2 is used (by user A) to encrypt the message, m, to create the signature, s, and it is used (by user B) to decrypt the signature, s. For example, to verify the signature, s, the verifier (user B) first checks that $r_2 < q$ (rejecting the signature, s, if this comparison fails) and then decrypts the signature,

---

[17] *See, e.g., Microsoft Computer Dictionary, Fourth Edition,* 1999 (Ex. 1031) ("**key** *n.* In encryption and digital signatures, a string of bits used for encrypting and decrypting information to be transmitted. Encryption commonly relies on two different types of keys, a public key known to more than one person (say, both the sender and the receiver) and a private key known only to one person (typically, the sender).")

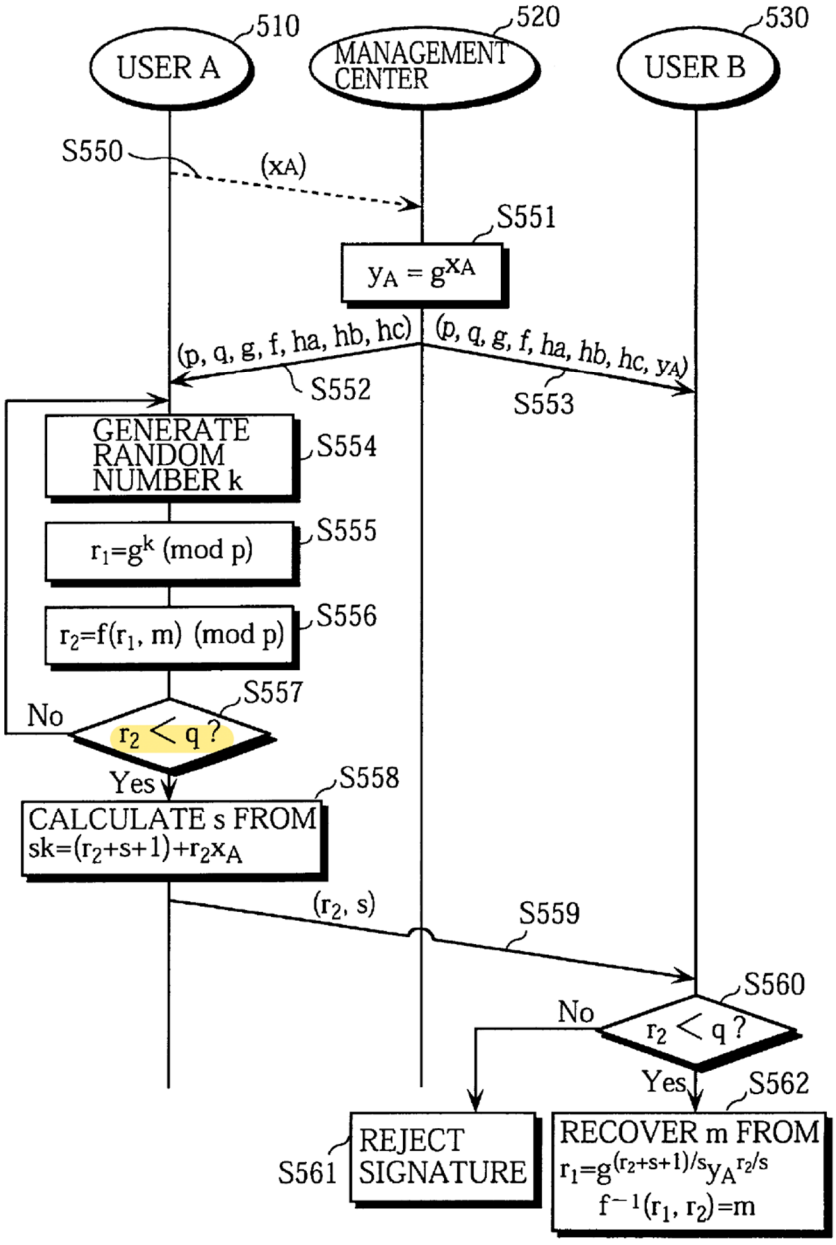
s, to get the original plaintext message, m, using the key, r_2, according to the following formula:

$$m = r_2 - g^{(r_2+s+1)/s} y_A^{r_2/s} \pmod{p}.$$

Id. at 8:64-9:25 (highlighting added).

155. This process is illustrated in Fig. 5 as follows:

FIG. 5



Miyaji at FIG. 5, (highlighting added).

156. Thus, a POSA would understand, for instance, that r_2 meets the requirements of the "key k" in limitations 1[a], 15[a] and 23[a]. For example, a POSA would understand that the key r_2 in Miyaji is a public key, e.g., since r_2 is sent

as plaintext from user A to user B along with the digital signature, s. Further, a POSA would also understand that the key r_2 in Miyaji is a short-term ephemeral key. For example, r_2 is an ephemeral key used for the communication session between user A and user B to communicate the digital signature, s, e.g., as illustrated in Fig. 5 excerpted above. Indeed, the '961 Patent also discloses both long-term keys and short-term ephemeral keys like that of Miyaji, which are used for digital signatures:

> **Public key cryptosystems may also be used to sign messages** to authenticate the author and/or the contents. In this case the sender will sign a message using his private key and a recipient can verify the message by applying the public key of the sender. If the received message and the recovered message correspond then the authenticity is verified.
>
> The public key cryptosystems rely on the intractability of the discrete log problem in finite field arithmetic, that is even when the generator α. and public key is known, it is computationally infeasible to obtain the corresponding private key. The security of such systems does therefore depend on the private key remaining secret. To mitigate the opportunity of disclosing the private key, protocols have been developed that use a pair of private keys and corresponding public keys, referred to as **long term and short term or ephemeral key pairs** respectively. The **ephemeral private key is generated at the start of each session between a pair of correspondents, usually by a random number generator**. The corresponding, **ephemeral public key is generated and the resultant key pair used in one of the possible operations described above**. The long-term public key is utilized to authenticate the correspondent through an appropriate protocol. Once the session is terminated, the ephemeral key is securely discarded and a new ephemeral key generated for a new session.

'961 Patent at 1:26-51 (emphasis added).

157. Thus, as explained above (and which I elaborate further below in my element-by-element analysis), a POSA would understand that r_2 is an ephemeral public key that is generated from the value "k" disclosed in Miyaji as being generated by a random number generator. That is, a POSA would understand the value "k" in Miyaji serves as an ephemeral private key corresponding to the ephemeral pubic key, r_2, and that r_2 is used to generate the digital signature, s. Therefore, a POSA would understand that r_2 is a "key k" as used in the preambles of claims 1, 15 and 23 of the '961 patent.

158. Zooming in on the loop in Fig. 5 of Miyaji and comparing this to Fig. 2 from the '961 Patent, which illustrates the steps of claims 1, 15 and 23 of the '961 patent, I find these two disclosures to be remarkably similar. Figure 5 from Miyaji and Figure 2 from the '961 patent reflect that the two methods, including generating a seed value k from RNG, performing a hash function on the seed value k to provide an output Hash H(k) stored in the masked message r_2, determining whether the output r_2 is less than order q prior to reducing mod q, accepting the output r_2 for use as said key k if the value of r_2 is less than said order q, rejecting r_2 if its value is not less than order q, repeating the method if it isn't, and providing the key k for use in

performing the cryptographic function if r_2 is accepted and the key k is equal to the output r_2, are nearly the same:

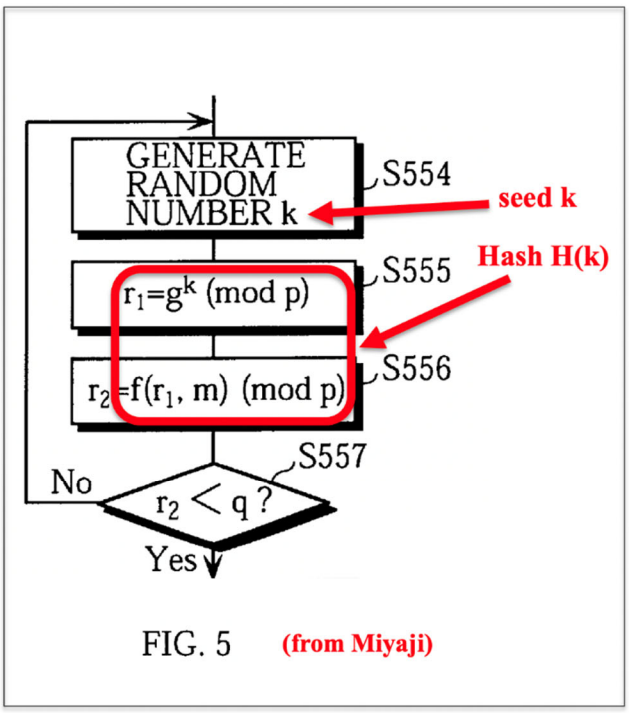


FIG. 5 (from Miyaji)

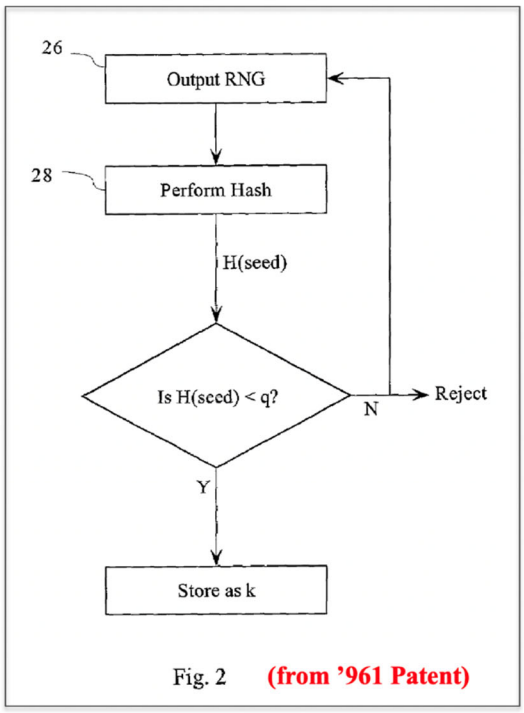


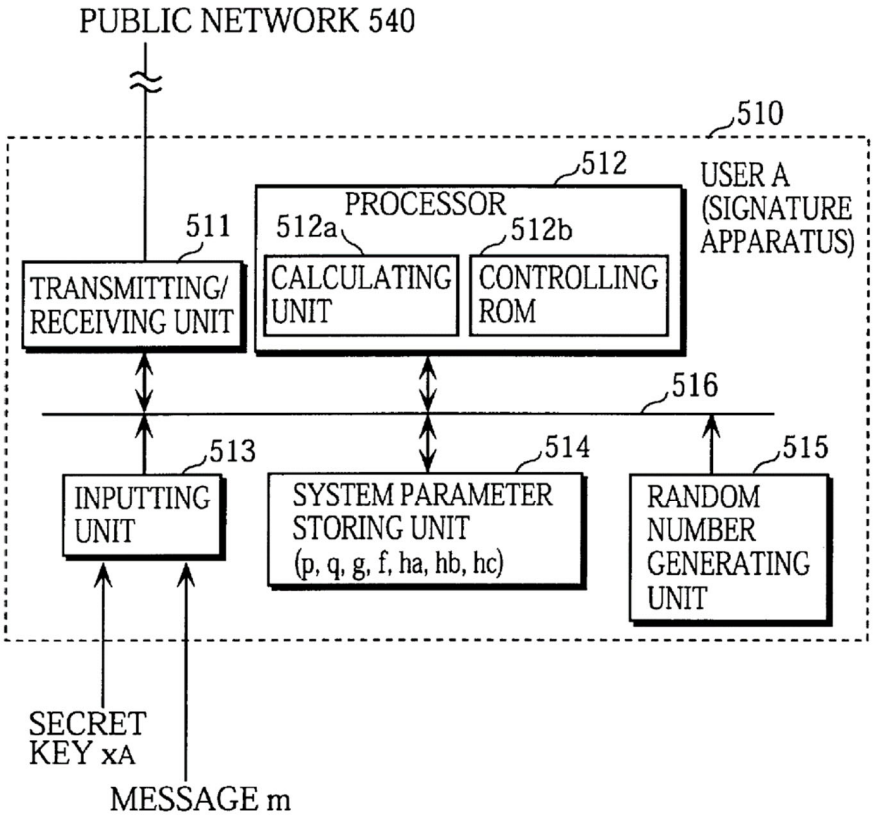
Fig. 2 (from '961 Patent)

Miyaji at FIG. 5; '961 Patent at Fig. 2 (annotations added in red).

159. Finally, to the extent that claims 15 and 23 add requirements for the use of a computer-readable medium or cryptographic unit for executing computer-executable and computer-readable instructions, Miyaji discloses those elements too. For instance, Miyaji discloses its cryptographic apparatus includes a processor 512 comprising a "ROM 512b storing a control program unique to the signature apparatus 510, and performs calculations and transmission control for signatures …, as well as controlling each component of the signature apparatus 510." Miyaji, at

6:12-16; *see also id.* at Abstract. Figure 2 reflects the processor and memory storing the control program, including the relevant instructions.

FIG. 2

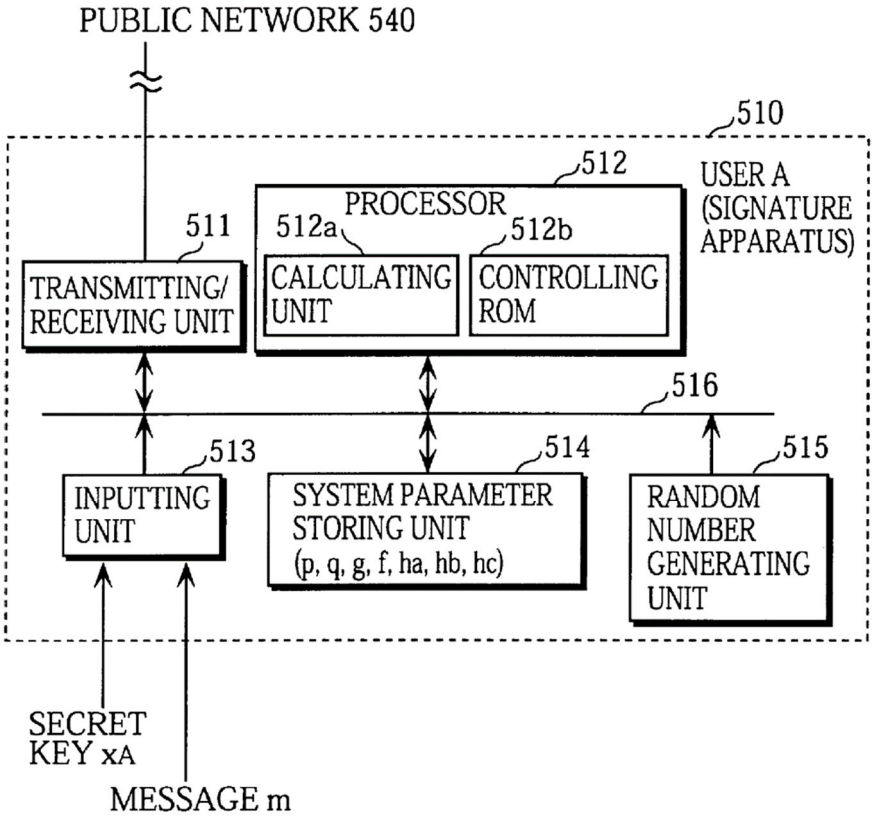


Miyaji, at FIG. 2. Further, Miyaji's cryptographic algorithm is performed "in accordance with the program in the controlling ROM 512b." *Id.*, at 8:11-30. A POSA would understand that the program controlled by ROM 512b necessarily requires both computer-readable and computer-executable instructions to function.

ii. [23][b]: "an arithmetic processor for"

160. Miyaji discloses an arithmetic processor, as required by independent claim 23. The processor illustrated in Figure 2 includes a "calculating unit," and thus is an arithmetic processor:

FIG. 2



Miyaji, at FIG. 2.

161. The calculating unit 512a also performs the "exponentiation modulo p through use of g and the random number k," and thus is an arithmetic unit. Miyaji, at 8:26-30.

iii. 1[b], 15[b], 23[c]: "generating a seed value SV from a random number generator;"

162. Miyaji discloses elements 1[b], 15[b], and 23[c], which all identically require "generating a seed value SV from a random number generator." For example, Miyaji uses the random number k as a seed for subsequent computations, as I explained above with respect to elements 1[a], 15[a], and 23[a] above. Again, looking at Figure 5, with respect to the random number, k, in Miyaji, which is used as a seed for subsequent computations, as explained above for elements 1[a], 15[a], and 23[a], Miyaji discloses the following regarding step S554:

> To place an order with the user B 530 for the product, the user A 510 first stores the received system parameters in the system parameter storing unit 514 and generates a random number k (512 bits long).
> Specifically, in accordance with the program in the controlling ROM 512b, the processor 512 stores the system parameters received via the transmitting/receiving unit 511 in the system parameter storing unit 514, has the **random number generating unit 515 generate the random number k** on GF(p), and acquires the random number k.

Miyaji at 8:12-21 (emphasis added).

163. Thus, the value k in Miyaji is disclosed as being generated by a "random number generator." Further, a POSA would understand that k is a "seed value," because it is used as an input to subsequent hash function, e.g., as required for a "seed value" disclosed in claim 1 of the '961 Patent. *See also, e.g.,* '961 Patent

at 2:40-44, 3:64-66, 4:6-10, and Fig. 5 excerpted above. Therefore, e.g., mapping the seed value k from Miyaji to the seed value SV from claim 1, Miyaji discloses elements 1[b], 15[b] and 23[c], "generating a seed value SV from a random number generator."

iv. 1[c], 15[c]. 23[d]:"performing a hash function H( ) on said seed value SV to provide an output H(SV);"

164. Miyaji discloses the requirement in each of claims 1, 15, and 23 for "performing a hash function H( ) on said seed value SV to provide an output H(SV)." For example, as explained above for element 1[a], 15[a] and 23[a], Miyaji discloses performing a hash function, H(k), on the seed value k to provide an output H(k), which is stored in the variable r_2. That is, a POSA would understand that Miyaji discloses the following derivation:

$$\begin{aligned} r_2 &= f(r_1, m) \\ &= r_1 + m \pmod p \\ &= g^k \pmod p + m \pmod p \\ &= g^k + m \pmod p \\ &= H(k). \end{aligned}$$

See, e.g., Miyaji at 8:23-35.

165. A POSA would understand the function, $f(r_1,m)$, which I denote as "H(k)," to be a "hash function," because it converts the input random value, k, which

Miyaji discloses to be 512 bits long, i.e., a number in the range from 0 to 2^{512}-1, to a number in the range from 0 to p-1, where p is a prime number 512 bits long. *See, e.g.,* Miyaji at 7:17-23, 8:12-15. Further, as I explain above, H(k) is itself the composition of two hash functions, namely, an exponential/discrete-logarithm hash function and a multiplicative hash function, as shown below in Fig. 5 of Miyaji.

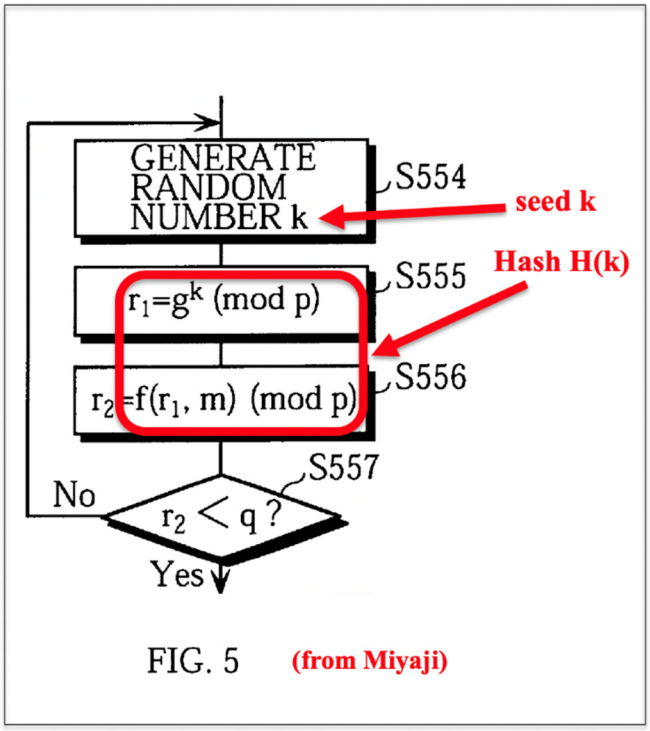


FIG. 5 (from Miyaji)

Miyaji at FIG. 5 (annotations added in red).

166. A POSA would understand that the composition of two hash functions is itself a hash function, as mentioned above, e.g., in ¶ 73. Thus, Miyaji discloses elements 1[c], 15[c] and 23[d], "performing a hash function H( ) on said seed value SV to provide an output H(SV)."

v. 1[d], 15[d], 23[e]: "determining whether said output H(SV) is less than said order q prior to reducing mod q;"

167. Miyaji discloses the requirement in each of independent claims 1, 15 and 23 for "determining whether said output H(SV) is less than said order q prior to reducing mod q." For example, Miyaji discloses determining whether r_2, which stores the output value, H(k), as I explain above, is less than q, which is the order of the element g in GF(p). *See, e.g.,* Miyaji at 7:17-23. In the flowchart of Figure 5, this step occurs at step S557:

> (Step **S557**)
> The processor **512** then compares r_2 and q according to the program in the controlling ROM **512***b*. If $q \leqq r_2$, steps **S554**~**S556** are repeated.

Miyaji at 8:42-45.

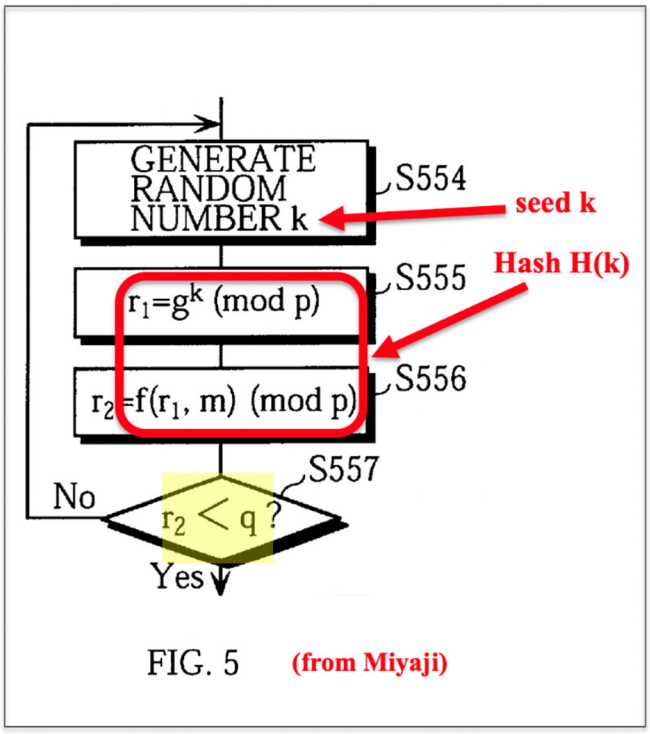


FIG. 5 (from Miyaji)

Miyaji at FIG. 5; '961 Patent at Fig. 2 (annotations added in red, highlighting in yellow).

168. Further, this step occurs prior to reducing mod q. For example, step S558, which comes after step S557, involves solving the following equation for the signature s mod q:

$$sk=(r_2+s+1)+r_2x_A \pmod q \qquad \text{(Equation 2.11)}$$

Id. at 8:53. *See also, e.g.,* Miyaji at 8:54-56 ("More specifically, the processor 512 receives the secret key x_A via the inputting unit 513 and has the calculating unit 512a solve s modulo q.").

169. Thus, Miyaji discloses elements 1[d], 15[d] and 23[e], "determining whether said output H(SV) is less than said order q prior to reducing mod q."

vi. 1[e], 15[f], 23[f]: "accepting said output H(SV) for use as said key k if the value of said output is less than said order q;"

170. Miyaji discloses the requirement in each of the independent claims 1, 15, and 23 for "accepting said output H(SV) for use as said key k if the value of said output is less than said order q." For example, Miyaji discloses that if r_2<q, where r_2 stores the output value, H(k), as explained above, the method in Miyaji accepts the key r_2 and continues to the next step to encrypt the message m using the key r_2 so as to produce the signature, s, e.g., as is disclosed as follows:

> (Step S**558**)
> If, on the other hand, r_2<q, the user A **510** solves s from the signature equation
>
> $$ha(r_2',s,\mathbf{1})k=hb(r_2',s,\mathbf{1})+hc(r_2',s,\mathbf{1})x_A(\text{mod } q) \qquad \text{(Equation 2.10)}$$
>
> that is
>
> $$sk=(r_2+s+1)+r_2x_A(\text{mod } q) \qquad \text{(Equation 2.11)}$$
>
> More specifically, the processor **512** receives the secret key x_A via the inputting unit **513** and has the calculating unit **512***a* solve s modulo q.

Miyaji at 8:46-46.

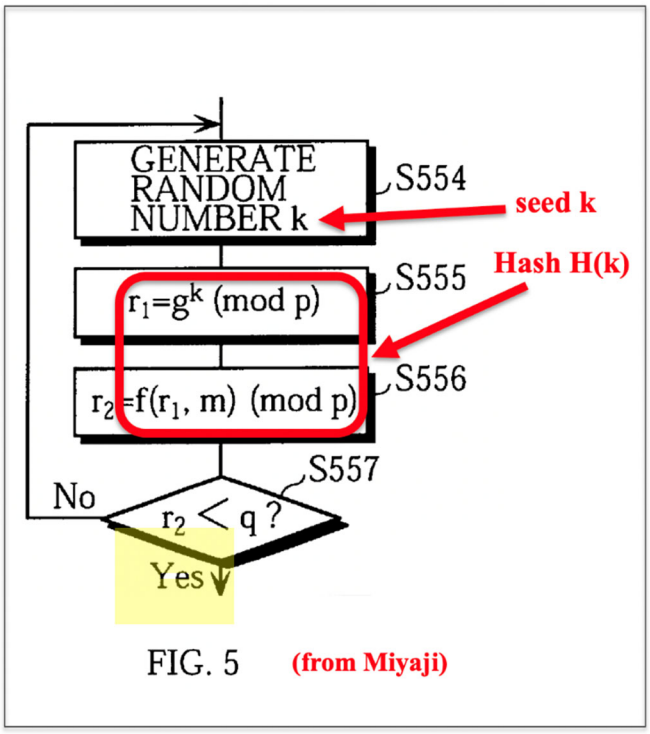


FIG. 5 (from Miyaji)

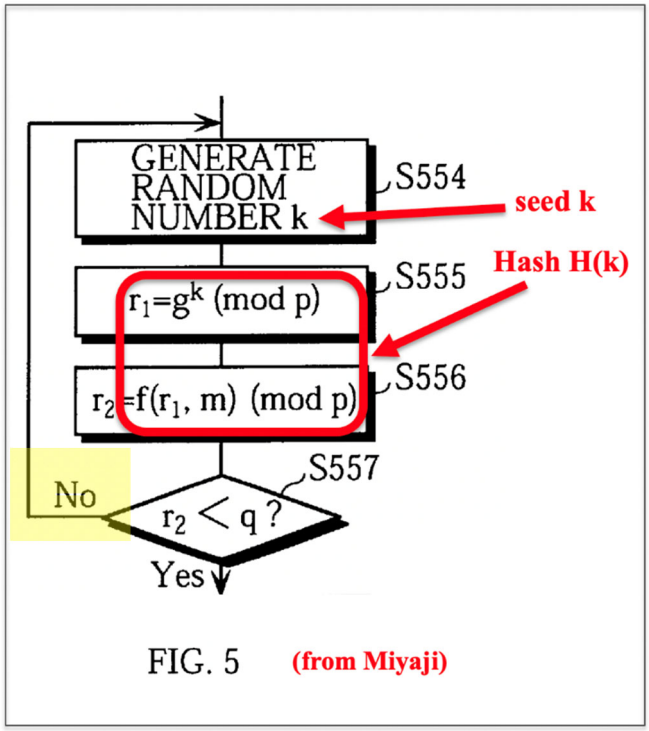
Miyaji at FIG. 5; '961 Patent at Fig. 2 (annotations added in red, highlighting in yellow).

171. In this way, Miyaji discloses accepting r_2, the result of the hash function, for use as a key to encrypt a message if the value is less than the order.

vii. 1[f], 15[f], 23[g]: "rejecting said output H(SV) as said key if said value is not less than said order q;"

172. Miyaji discloses the identical requirement in each of claims 1, 15 and 23 for "rejecting said output H(SV) as said key if said value is not less than said order q." For example, if r_2, which stores the output value, H(k), as explained above, is not less than q, then the value stored in r_2 is rejected and the steps for computing the key r_2 are repeated, starting from the generation of the random number k. *See,*

e.g., Miyaji at 8:42-45 ("(Step 557). The processor 512 then compares r_2 and q according to the program in the controlling ROM 512b. *Id.* Myiyaji states that "[i]f $q \leq r_2$, steps S554-S556 are repeated." *Id.* This process is also shown in the flowchart for Figure 5, as depicted in the highlighted "No" choice, which I excerpt below:



Miyaji at FIG. 5; '961 Patent at Fig. 2 (annotations added in red, highlighting in yellow).

viii. 1[g], 15[g], 23[h]: "if said output H(SV) is rejected, repeating said method; and"

173. Miyaji discloses the requirement in each of the independent claims that "if said output H(SV) is rejected, repeating said method." For example, as shown above for elements 1[f], 15[f] and 23[g], if r_2, which stores the output value, H(k),

as explained above, is not less than q, then the value stored in r_2 is rejected and the steps for computing the key r_2 are repeated, starting from the generation of the random number k. *See, e.g.,* Miyaji at 8:42-45 ("(Step 557). As Miyaji notes, the processor 512 then compares r_2 and q according to the program in the controlling ROM 512b, and if "$q \leq r_2$, **steps S554-S556 are repeated**." *Id.* (emphasis added). This step is again illustrated by Figure 5, as annotated below:

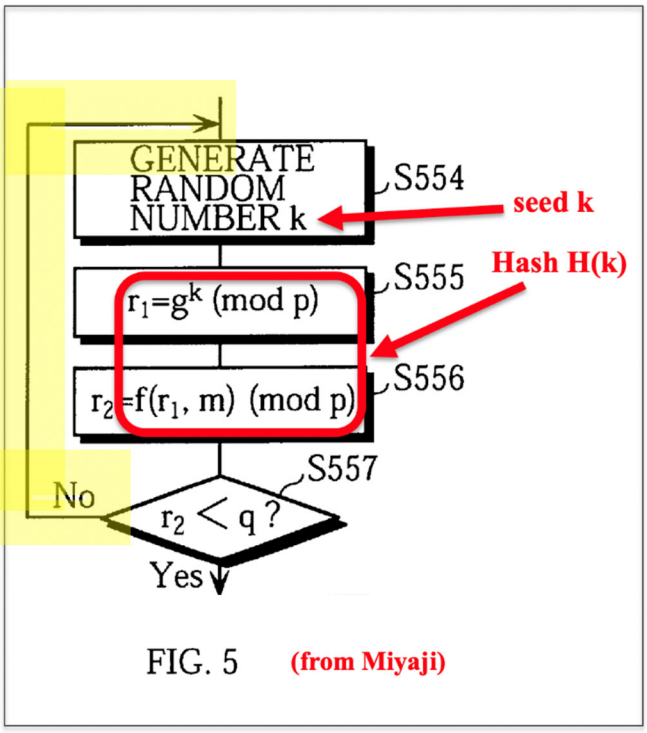


FIG. 5 **(from Miyaji)**

Miyaji at FIG. 5; '961 Patent at Fig. 2 (annotations added in red, highlighting in yellow).

ix. 1[h], 15[h], 23[i]: "if said output H(SV) is accepted, providing said key k for use in performing said

cryptographic function, wherein said key k is equal to said output H(SV).”

174. Miyaji discloses the final element of each of claims 1, 15 and 23 that “if said output H(SV) is accepted, providing said key k for use in performing said cryptographic function, wherein said key k is equal to said output H(SV).” For example, as explained above for elements 1[e], 15[e] and 23[f], if the value stored in r_2 is accepted, then the key r_2, which stores the output H(k), is used to produce the signature s, as shown below for step S558. In Miyaji, producing the signature s is a cryptographic function, because the next step, S559, involves user A sending to user B the *ciphertext* (r_2,s), consisting of the key, r_2, and signature, s:

(Step S**558**)
If, on the other hand, r_2<q, the user A **510** solves s from the signature equation

$$ha(r_2',s,\mathbf{1})k=hb(r_2',s,\mathbf{1})+hc(r_2',s,\mathbf{1})x_A(\text{mod } q) \qquad \text{(Equation 2.10)}$$

that is

$$sk=(r_2+s+1)+r_2x_A(\text{mod } q) \qquad \text{(Equation 2.11)}$$

More specifically, the processor **512** receives the secret key x_A via the inputting unit **513** and has the calculating unit **512***a* solve s modulo q.
(Step S**559**)
Lastly, the user A **510** sends a ciphertext (r_2,s) to the user B **530**.
Specifically, the processor **512** sends the ciphertext (r_2,s) to the user B **530** via the transmitting/receiving unit **511**.

Miyaji at 8:46-61.

175. Therefore, Miyaji anticipates claim 1 of the ’961 Patent.

x. Claims 2, 16, and 24

176. Claims 2, 16, and 24 are all dependent claims, and identically require: "The method [computer readable medium/cryptographic unit] of claim 1 [15/23] wherein another seed value is generated by said random number generator if said output is rejected." In my opinion, claims 2, 15, and 23 of the '961 patent all are anticipated by Miyaji. I incorporate by reference my analysis for claims 1, 15, and 23. For example, I explain above that Miyaji discloses each limitation of claims 1, 15, and 23. In addition, Miyaji discloses that in the case when the output r_2, which stores the output, H(k), as I explain above for claim elements 1[g], 15[g] and 23[h], is rejected, the method of Miyaji repeats the process for computing r_2 (i.e., steps S554-S556) starting from the step (S554) of producing another seed value k according to a random number generator:

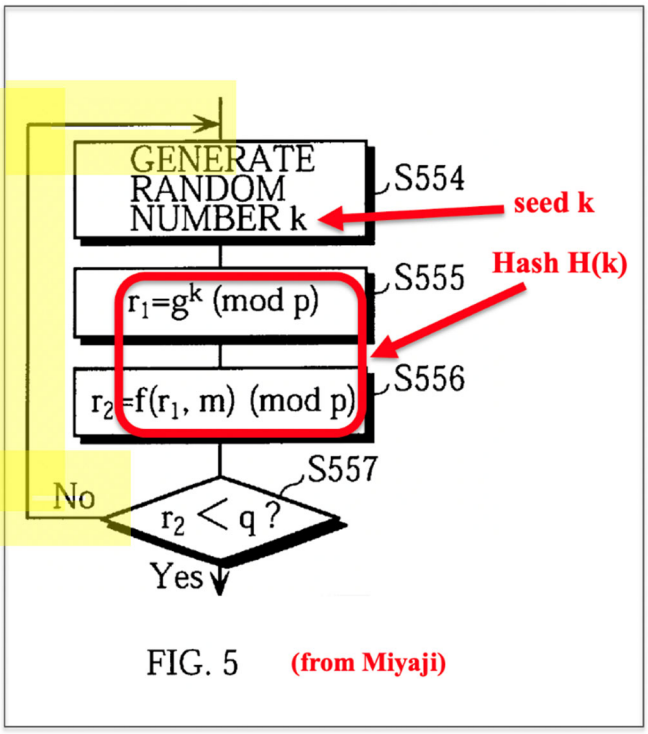


FIG. 5 (from Miyaji)

Miyaji at FIG. 5; '961 Patent at Fig. 2 (annotations added in red, highlighting in yellow); *see also, e.g.,* Miyaji at 8:11-15 ("(Step S554) To place an order with the user B 530 for the product, the user A 510 first stores the received system parameters in the system parameter storing unit 514 and generates a random number k (512 bits long)."); *id.* 8:42-45 ("(Step S557) The processor 512 then compares r_2 and q according to the program in the controlling ROM 512b. If $q \leq r_2$, steps S554-S556 are repeated."). Thus, Miyaji discloses the requirements in each of claims 2, 16, and 24 "wherein another seed value is generated by said random number generator if said output is rejected."

xi. Claims 3, 17, and 25

177. Claims 3, 17, and 25 are all dependent claims that follow from the independent claims and identically require: "wherein the step of accepting said output as a key includes a further step of storing said key." In my opinion, claims 3, 17, and 25 of the '961 patent all are anticipated by Miyaji. I incorporate by reference my analysis for claims 1, 15, and 23. For example, I explain above that Miyaji discloses each limitation of claims 1, 15, and 23. In addition, as I explain above for elements 1[d], 15[d] and 23[e], and incorporate by reference here, Miyaji discloses storing the output, H(k), in the variable r_2. Further, Miyaji employs a "system storing unit 514" that "is a RAM or the like" which "temporarily stores system parameters downloaded from the management center 520." Miyaji at 6:18-20. The system parameters are those "necessary for the message recovery scheme" used in Miyaji, and necessarily include the output H(k). *Id.* at 6:20-23. Thus, Miyaji discloses the requirements in each of claims 3, 17, and 25 "wherein the step of accepting said output as a key includes a further step of storing said key."

xii. Claims 4, 18, and 26

178. Claims 4, 18, and 26 all are dependent claims that identically require "wherein said key is used for generation of a public key." In my opinion, claims 4, 18, and 26 of the '961 patent all are disclosed by Miyaji. I incorporate by reference

my analysis for claims 1, 15, and 23. For example, I explain above that Miyaji discloses each limitation of claims 1, 15, and 23. In addition, Miyaji discloses storing the output, H(k), in the key, r_2, which is used by user B to decrypt the signature, s. *See, e.g.,* Miyaji at 8:62-9:34.

179. Miyaji describes r_2 as a "masked message" that is generated from the "commitment" r_1 (which is generated as an exponentiation using a random nonce, k) and the message "m." Miyaji at 4:23-26, 5:5-6; claim 1. Miyaji also identifies r_2 as being part of the "ciphertext (r_2,s)" that is sent to the user B for decryption of the signature, s. *Id.* at Abstract, 2:12-13, 2:31-32.

180. Thus, a POSA would recognize that the key r_2 is public (e.g., it is sent to user B from user A), and that it has an associated private key, k. For example, the "masked message" or key r_2 is an ephemeral public key used by User B to decrypt a digital signature, s, to get the resulting original plaintext message, m, which a POSA would recognize is a property of a public key. *See, e.g.,* the '961 patent at 1:26-51 (excerpted above at ¶ 156). For instance, Miyaji discloses that user B recovers the message, m, using r_2 as follows:

(Step S**562**)
If r_2<q, the user B **530** finds r_1 by computing

$$r_1 = g^k = g^{(r2+s+1)/s} y_A^{r2/s} \pmod{p} \quad \text{(Equation 2.12)}$$

and recovers the message m according to

$$f^{-1}(r_1,r_2)=m(\text{mod } p) \quad \text{(Equation 2.13)}$$

which can be expanded to

$$m = f^{-1}(r_1, r_2) = r_2 - r_1 = r_2 - g^k = r_2 - g^{(r2+s+1)/s} y_A^{r2/s} \quad \text{(Equation 2.14)}$$

Miyaji at 9:4-25 (highlighting added).

181. Thus, a POSA would understand r_2 to be a public key. That is, in my opinion, the (short-term) ephemeral public key, r_2, meets the requirements for a "public key" of the claim. *See also* discussion *supra* at ¶¶ 156-157.

182. Therefore, in my opinion, claims 4, 18, and 26 of the '961 patent all are disclosed by Miyaji.

B. OBVIOUSNESS

i. Miyaji in View of Ordinary Skill in the Art

183. In my opinion, Miyaji in view of ordinary skill in the art renders obvious each of claims 1-6, 15-20, and 23-86 of the '961 patent.

ii. Claims 1, 15, and 23

184. In my opinion, claims 1, 15, and 23 of the '961 patent are rendered obvious by Miyaji, e.g., in light of the admitted knowledge of a POSA, as acknowledged by the '961 patent's specification. I incorporate by reference my analysis above that claims 1, 15, and 23 of the '961 patent are anticipated by Miyaji. For example, as I explain above, Miyaji teaches generating a seed, k (which can be mapped to SV), from a random number generator, applying a hash function to get a result, r_2, rejecting r_2 if it is not less than q, and repeating the method until r_2 is less than q, and then using r_2 as an ephemeral public key for a signature, s, for a message, m, i.e., using a method that discloses the elements of each of claims 1, 15 and 23 of the '961 patent.

185. Although Miyaji never explicitly states that r_2 is a public key, Miyaji nonetheless describes r_2 as a "masked message" that is generated from the "commitment" r_1 (which is generated from a random nonce, k) and the message "m." Miyaji at 4:23-26, 5:5-6; claim 1. Miyaji also identifies r_2 as being part of the "ciphertext (r_2,s)" that is sent to the user B for decryption. *Id.* Abstract, 2:12-13, 2:31-32.

186. As further reflected by my analysis of claims 4, 18, and 26 above, a POSA would nonetheless readily appreciate that the "masked message" or key r_2 is

public (e.g., it is sent to user B from user A), that it has an associated private key, k, and that it is used to decrypt the signature, s. In any event, even if it somehow was ambiguous whether r_2 actually is a public key given that Miyaji never explicitly states as much (something I don't think any POSA would conclude), Miyaji in view of a POSA still would render the independent and dependent claims obvious. That is because the '961 patent acknowledges that a POSA would be aware of the DSA signature standard and the bias in how the (NIST) FIPS 186 standard computes the key k used in DSA (e.g., see the '961 patent at 1:52-64 and 2:32-55).

187. A POSA would find it obvious to use the method disclosed in Miyaji to generate the same public key k of DSA, e.g., by setting $k=r_2$, where r_2 is an ephemeral public key for a null message, m=0, according to the method of Miyaji. A POSA with an understanding of the acknowledged prior art of the '961 patent would understand there would be an expectation of success in using Miyaji in combination with the DSA key generation method from FIPS 186 described as ordinary skill in the art in the '961 patent. In fact, in my opinion, that is what Miyaji already is doing.

188. In substituting the public key k for masked message r_2 a POSA would have the expectation of success of eliminating the bias that otherwise occurs when generating DSA's key k. Miyaji's masked message r_2 already performs the same function as a public key k like that in DSA. Miyaji also uses the well-known

rejection sampling technique that eliminates bias to generate r_2. Therefore, a POSA would recognize that this combination of Miyaji and ordinary skill in the art to substitute the public key k for masked message r_2 produces a random key, k, like that already in use in DSA that is uniformly distributed in the range from 1 to q-1, thereby eliminating the bias in the way that the FIPS 186-1 and 186-2 documents describe. Further, this combination of Miyaji and ordinary skill in the art would disclose all the other elements of claim 1, 15, and 23 of the '961 patent for the same reasons I give above for how Miyaji anticipates claims 1, 15, and 23 of the '961 patent.

iii. Claims 2, 16, and 24

189. Claims 2, 16, and 23 are dependent claims that each require: "wherein another seed value is generated by said random number generator if said output is rejected." In my opinion, to the extent that claims 2, 16, and 24 somehow are not anticipated by Miyaji, each of these claims are rendered obvious by Miyaji, e.g., in light of the admitted knowledge of a POSA. To that end, I incorporate by reference my analysis in the preceding section for why claims 1, 15, and 23 are rendered obvious by Miyaji, e.g., in light of the admitted knowledge of a POSA.

190. In addition, Miyaji discloses that in the case when the output r_2, which stores the output, H(k), as I explain above for why Miyaji discloses claim elements 1[g]. 15[g] and 23[h], is rejected, the method of Miyaji repeats the process for

computing r_2 (i.e., steps S554-S556) starting from the step (S554) of producing another seed value k according to a random number generator:

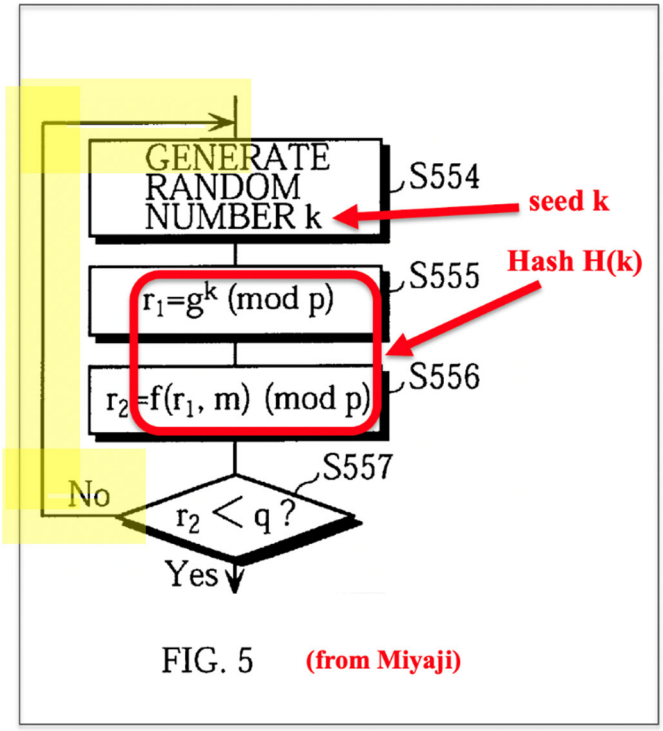


FIG. 5 **(from Miyaji)**

Miyaji at FIG. 5; '961 Patent at Fig. 2 (annotations added in red, highlighting in yellow); *see also, e.g.,* Miyaji at 8:11-15 ("(Step S554) To place an order with the user B 530 for the product, the user A 510 first stores the received system parameters in the system parameter storing unit 514 and generates a random number k (512 bits long)."); *id.* at 8:42-45 ("(Step S557) The processor 512 then compares r_2 and q according to the program in the controlling ROM 512b. If $q \leq r_2$, steps S554-S556 are repeated.").

191. Further, for the same reasons as I explain above for why Miyaji renders claims 1, 15, and 23 of the '961 patent obvious, a POSA would find it obvious to use the method disclosed in Miyaji to generate the key k of DSA, e.g., by setting $k=r_2$, where r_2 is an ephemeral public key for a null message, m=0, according to the method of Miyaji. A POSA would understand this combination would have an expectation of success in eliminating a bias reflected by the already known prior art DSA key generation method from FIPS 186 described as admitted knowledge in the art in the '961 patent (e.g., see the '961 patent at 1:52-64 and 2:32-55). A POSA would have the expectation of success of eliminating the bias in the generation of DSA's key k, since the method of Miyaji uses the well-known rejection sampling technique to generate r_2, thereby eliminating the bias in the way that the FIPS 186-1 and 186-2 documents describe. That is, a POSA would recognize that the combination of Miyaji and ordinary skill in the art provides a random key, k, for use in DSA that is uniformly distributed in the range from 1 to q-1. Further, this combination would disclose all the other elements of the independent claims incorporated into claims 2, 16, and 24 of the '961 patent, as I explain above.

192. Thus, claims 2, 16, and 24 which all require "wherein another seed value is generated by said random number generator if said output is rejected," are

each independently rendered obvious by Miyaji, e.g., in light of the admitted knowledge of a POSA.

iv. Claims 3, 17, and 25

193. Claims 3, 17, and 25 all are dependent claims that require: "wherein the step of accepting said output as a key includes a further step of storing said key." Once again, to the extent there is any ambiguity whether masked message r_2 is a public key (which in my opinion there shouldn't be), it is nonetheless my opinion that claims 3, 17, and 25 of the '961 patent are each rendered obvious by Miyaji, e.g., in light of the admitted knowledge of a POSA. I incorporate by reference my analysis for why claims 1, 15, and 23 are rendered obvious by Miyaji, e.g., in light of the admitted knowledge of a POSA. In addition, as I explain above for why Miyaji discloses elements 1[d], 15[d] and 23[e], and incorporate by reference here, Miyaji discloses storing the output, H(k), in the variable r_2.

194. Further, for the same reasons as I explain above for why Miyaji in combination with ordinary skill in the art renders claims 1, 15, and 23 of the '961 patent obvious, a POSA also would find it obvious to use the method disclosed in Miyaji to generate a key like the key k of DSA, e.g., by setting $k=r_2$, where r_2 is an ephemeral public key for a null message, m=0, according to the method of Miyaji. A POSA would appreciate that this application of ordinary knowledge in

combination with Miyaji would produce an expectation of success in storing a key, because by using admitted knowledge in the art in key generation like that described in the '961 patent from the known DSA key generation method from FIPS 186 with Miyaji a POSA will necessarily store a key efficiently. A POSA would expect success from the combination of this use of ordinary knowledge with Miyaji by eliminating the bias in the generation of DSA's key k, since the method of Miyaji uses the well-known rejection sampling technique to generate r_2, thereby eliminating the bias in the way that the FIPS 186-1 and 186-2 documents describe. That is, a POSA would recognize that this combination of Miyaji and ordinary skill in the art provides a random key, k, for use in a protocol like DSA that is uniformly distributed in the range from 1 to q-1. Further, this combination would disclose all the other elements of claims 3, 17, and 25 of the '961 patent required by the independent claims, as I explain above.

195. Thus, Miyaji discloses the requirement in each of claims 3, 17, and 25 "wherein the step of accepting said output as a key includes a further step of storing said key."

v. Claims 4, 18, and 26

196. Claims 4, 18, and 26 each are dependent upon the independent claims 1, 15, and 23, and require: "wherein said key is used for generation of a public key."

In my opinion, claims 4, 18, and 26 of the '961 patent each are rendered obvious by Miyaji, e.g., in light of the admitted knowledge of a POSA. I incorporate by reference my analysis for why Miyaji discloses each element of claims 1, 15, and 23 above.

197. As I explain above in conjunction why I believe Miyaji actually anticipates claims 4, 18, and 26, a POSA would recognize that the key r_2 is public (e.g., it is sent to user B from user A), that it has an associated private key, k, and that it is used to decrypt the digital signature, s. For example, the key r_2 is an ephemeral public key used by User B to decrypt a digital signature, s, to get the resulting original plaintext message, m, which a POSA would recognize is a property of a public key. *See, e.g.,* '961 patent at 1:26-51 (excerpted above at ¶ 156). For instance, Miyaji discloses that user B recovers the message, m, using r_2 as follows:

(Step S**562**)
If r_2<q, the user B **530** finds r_1 by computing

$$r_1 = g^k = g^{(r2+s+1)/s} y_A^{r2/s} \pmod{p} \quad \text{(Equation 2.12)}$$

and recovers the message m according to

$$f^{-1}(r_1,r_2)=m(\text{mod } p) \quad \text{(Equation 2.13)}$$

which can be expanded to

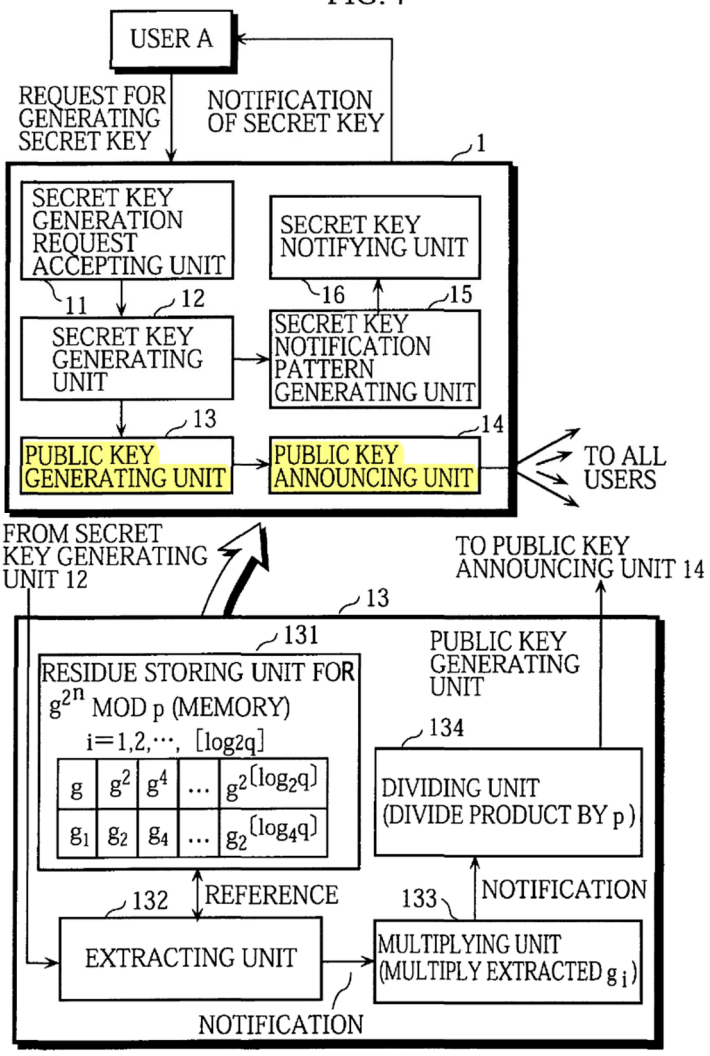
$$m = f^{-1}(r_1, r_2) = r_2 - r_1 = r_2 - g^k = r_2 - g^{(r2+s+1)/s} y_A^{r2/s} \quad \text{(Equation 2.14)}$$

Miyaji at 9:4-25 (highlighting added).

198. Thus, a POSA would understand r_2 to be a public key, since it is a (short-term) ephemeral public key. Thus, r_2, meets the requirements for a "public key" of the claim. *See also* discussion *supra* at ¶¶ 156-157.

199. Further, even if r_2 somehow were not already understood by a POSA to be a public key, it still would be obvious to use the method of Miyaji with ordinary skill to generate a (long-term) public key. Miyaji discloses a public key generating unit and a public key announcing unit, as reflected by Figure 7:

FIG. 7



.

Ex. Miyaji at FIG. 7 (highlighting added). Significantly, the '961 patent acknowledges that ordinary skill of a POSA includes the knowledge that in the ECDSA elliptic curve digital signature algorithm, a public key kP is generated from a random number key in the range from 0 to n-1. *See* '961 patent, at 2:2-11. Yet, Miyaji also discloses elliptic-curve public keys of this form:

> The message recovery signature scheme in this system is a public key cryptosystem that uses the discrete logarithm problem as the founding principle for the security, and is based on operations over an elliptic curve E(GF(p)) defined on a finite field GF(p) where p is a prime number.
>
> An elliptic curve referred to here is a function that is generally expressed as $y^2=x^3+ax+b$, with E(GF(p)) denoting the set of points (x,y) that are elements of GF(p) and are present on the elliptic curve. The discrete logarithm problem based on the elliptic curve E(GE(p)) is as follows. Let Q and G be elements of E(GF(p)). The problem is to find a natural number d that satisfies the relationship
>
> $Q=dG$ (Equation 3.1)

Miyaji at 11:45-58.

> <Public Key Generation>
>
> First, the management center **520** uses the user A's secret key x_A which has been informed by the user A **510** beforehand (S**570**) to generate a public key y_A of the user A **510** according to
>
> $Y_A=x_AG$ (Equation 3.6)
>
> (S**571**). The management center **520** then reveals the system parameters (p, E, G, ha, hb, hc) to the user A **510** and user B **530**, as well as informing the user B **530** of the user A's public key Y_A (S**572** and S**573**).

Miyaji at 12:20-29.

200. Thus, even if masked message r_2 somehow was not already recognized by a POSA to be a public key (which to me it clearly is), in my opinion it still would be obvious to a POSA to use the method of Miyaji for generating a random value k

in a given range 0 to n-1, e.g., by setting $k=r_2$ for a null message, m=0, and to then apply ordinary skill in the art and use this value k to generate the public key kP in ECDSA. A POSA would understand the benefit of generating a random value that is uniform in a given interval, and would apply this ordinary skill to take advantage of the public-key generation method of ECDSA.

201. Further, this use of ordinary knowledge of one method for generating a random key in ECDSA with Miyaji's masked message key r_2 to perform the same function as Miyaji's produces better uniformity in the result. Hence, there would be an expectation of success that this substitution of EDCSA's technique for generating a key via ordinary knowledge eliminates any bias in generating the random key used in ECDSA.

vi. Claims 5, 19, and 27

202. Claims 5, 19, and 27 each are dependent claims that identically require: "wherein said order q is a prime number represented by a bit string of predetermined length L." In my opinion, claims 5, 19, and 27 of the '961 patent each are rendered obvious by Miyaji, e.g., in light of the admitted knowledge of a POSA. For example, I incorporate by reference my analysis above for why I believe Miyaji anticipates claims 1, 15, and 23. I further incorporate by reference my analysis above for why

I believe that even if Miyaji does not anticipate claims 1, 15, and 23, the combination of Miyaji and ordinary skill in the art nonetheless renders those claims obvious.

203. In addition, Miyaji discloses that its method is for a group with order q. *See*, *e.g.*, Miyaji at 2:28-29 ("Note that (mod p) and (mod q) denote operations modulo p and q respectively"). As I explain above both for anticipation and obviousness of claims 1, 15, and 23, and incorporate by reference here, Miyaji satisfies all the limitations of claims 1, 15, and 23 regardless of whether or not the masked message r_2 is recognized (as it should be to a POSA) to be a key.

204. As a result, it also would be obvious to a POSA using nothing more than ordinary skill in the art that the order of this group q can be a prime number. Such ordinary knowledge is exemplified by the POSA's admitted knowledge of DSA and the mathematics behind DSA, including Cauchy's theorem. *See* '961 patent, 1:52-62 ("Some of the more popular protocols for signature are the ElGamal family of signature schemes such as the Digital Signature Algorithm or DSA."); *see also id.* at 1:63-2:14 (explaining signature generation in DSA). For instance, a POSA would know that DSA requires an element of order q where q is a prime number, which supports its security, and that, as I explain above in ¶ 70, Cauchy's theorem guarantees success in finding such an element; hence, a POSA would find it useful for q to be a prime number. For example, as I explain above for the

obviousness of claim 1, a POSA would find it obvious to use the method of Miyaji to generate a key like the k of DSA, which is required to be less than a number, q, which is prime.

205. Further, Miyaji discloses that q is represented by a bit string of predetermined length L. For example, Miyaji discloses that q is represented by a bit string that is 256 bits long. *See, e.g.,* Miyaji at 7:17-23 ("The message recovery signature scheme used in this system is a public key cryptosystem that uses the discrete logarithm problem as the founding principle for the security, and is based on operations over a finite field GF(p) where p (512 bits long) is a prime number, g is an element, and q (256 bits long) is the order of g.").

206. In my opinion, a POSA already would recognize the advantages of, and it would be obvious for a POSA to try, the above substitution of an element with an order q, wherein q is represented by a bit string of predetermined length L, with an element with an order q, wherein q is a prime number represented by a bit string of predetermined length L. There would be an expectation of success from this combination that immediately would be recognized by a POSA to enhance the security of the method, as I explain above in ¶ 71. Further, there would be a reasonable expectation of success that this combination would eliminate the bias of the method disclosed in FIPS 186-1 and 186-2.

207. Thus, in my opinion, claims 5, 19, and 27 of the '961 patent are rendered obvious by Miyaji, e.g., in light of the admitted knowledge of a POSA.

vii. Claims 6, 20, and 28

208. Claims 6, 20, and 28 are each multiply dependent claims that follow from claims 5, 19, and 27. Each of claims 6, 20, and 28 each require: "wherein said output from said hash function is a bit string of predetermined length L." In my opinion, claims 6, 20, and 28 of the '961 patent are rendered obvious by Miyaji, e.g., in light of the admitted knowledge of a POSA. As I explain above both for anticipation and obviousness of claims 1, 15, and 23, and incorporate by reference here, Miyaji satisfies all the limitations of claims 1, 15, and 23 regardless of whether or not the masked message r_2 is recognized (as it should be to a POSA) to be a key. For the same reasons, I incorporate by reference my analysis above why I believe that claims 5, 19, and 27 also are rendered obvious by the combination of Miyaji and ordinary skill.

209. In addition, Miyaji discloses that its hash function, "f", produces the output that is represented by a bit string that is a predetermined length, or 512 bits long. The output is produced from the seed "k" and stored in r_2 is computed as mod p, where p is represented by a bit string that is 512 bits long. *See, e.g.,* Miyaji at 7:17-23 ("The message recovery signature scheme used in this system is a public

key cryptosystem that uses the discrete logarithm problem as the founding principle for the security, and is based on operations over a finite field GF(p) where p (512 bits long) is a prime number, g is an element, and q (256 bits long) is the order of g."), and 8:11-41. Thus, the same POSA who would appreciate that the combination of Miyaji and ordinary skill produces an order q that is a prime number represented by a bit string of predetermined length L as required by claims 5, 19, and 27 of the '961 patent, would necessarily further understand that this output p and r_2 are also represented by a bit string of predetermined length L, where L is 512 bits, providing the benefits of security in using a large prime number.

210. Thus, in my opinion, claims 6, 20, and 28 of the '961 patent all are rendered obvious by Miyaji, e.g., in light of the admitted knowledge of a POSA. patent.

C. Motivation to Combine

211. A POSA would be motivated combine the teachings of Miyaji with the acknowledged ordinary skill of a POSA as reflected by the skill taught in the '961 patent. A POSA would have been motivated to combine Miyaji's rejection sampling technique employing the public key k of DSS or the public key kP in ECDSA disclosed in the '961 patent, since a POSA would have a reasonable expectation of success in doing so, e.g., to avoid any bias in generating such a key.

A POSA would recognize this combination of Miyaji and ordinary skill offers predictable results.

212. For instance, a POSA would recognize that it would be beneficial to employ the public key k from DSS, which generated bias in its key as described in the '961 patent with Miyaji's rejection sampling technique using the masked message r_2 from Miyaji (which already is a public key), with a null message[18], e.g., setting k=r_2 for DSS, because doing so reduces bias in the selection of the generated key of DSS. The same would be true for the combination of ECDSA with Miyaji. To a POSA, using the rejection sampling technique and r_2 of Miyagi and the known techniques and generation of a public key from DSS would simply require substitution of one known element (generating of a public key using the techniques described as prior art in the '961 patent) for another (the rejection sampling technique that reduces bias described in Miyaji). This combination would have additionally constituted the use of a known technique (generation of a public key k or kP from DSS and/or ECDSA) to improve an algorithm directed to reducing bias,

[18] A POSA would be aware that there is no message in a key generation protocol; hence, a POSA would use a null message, M=0, in employing the method of Miyaji for key generation.

such that the known technique would function in the same way as before, with predictable results.

D. Obviousness by Bellare in View of LEDA

213. In my opinion, claims 1-6, 15-20, and 23-28 of the '961 patent are rendered obvious by Bellare in light of LEDA. For example, as I explain in my discussion of the motivation to combine Bellare and LEDA, a POSA would recognize the combination of Bellare with LEDA to have two embodiments, either of which, in my opinion, and as I explain below in my element-by-element analysis, renders the above-referenced claims obvious:

- Bellare and LEDA, using the ***truncated linear congruential generator*** of Bellare for internal_source() in LEDA and using the (otherwise unchanged) random number generator taught in LEDA at p. 93 to generate the random number k in "the scheme" taught in Bellare.
- Bellare and LEDA, using ***SHA-1,*** as recommended by Bellare, for internal_source() in LEDA and using the (otherwise unchanged) random number generator taught in LEDA at p. 93 to generate the random number k in "the scheme" taught in Bellare.

i. Claims 1, 15, and 23

214. In my opinion, claims 1, 15, and 23 of the '961 Patent are rendered obvious by Bellare in light of LEDA.

> **1[a]: "A method of generating a key k for use in a cryptographic function performed over a group of order q, said method including the steps of:"**
>
> **15[a]: "A computer readable medium comprising computer executable instructions for generating a key k for use in a cryptographic function performed over a group of order q, said instructions including instructions for:**
>
> **23[a]: "A cryptographic unit configured for generating a key k for use in a cryptographic function performed over a group of order q, said cryptographic unit having access to computer readable instructions and comprises:"**

215. To the degree that the preambles of claims 1, 15, or 23 are limiting, all of which require "generating a key k for use in a cryptographic function performed over a group of order q," they are disclosed and rendered obvious by Bellare in view of LEDA.

216. For example, Bellare discloses a method of generating a key for use in a cryptographic function, which it refers to as "the scheme" for the Digital Signature Algorithm (DSA), which a POSA would understand to be a cryptographic function. Bellare also discloses that "the scheme" generates a key k for use in the DSA

cryptographic function, and that this function is performed over the group $\mathbf{Z}_q$, which has order q, e.g., as follows:

> THE SCHEME. The scheme uses the following parameters: a prime number p, a prime number q which divides $p-1$ and an element $g \in Z_p^*$ of order q. (Chosen as $g = h^{(p-1)/q}$ where h is a generator of the cyclic group Z_p^*). These parameters may be common to all users of the signature scheme and we will consider them as fixed in the rest of the paper. The standard asks that $2^{159} < q < 2^{160}$ and $p > 2^{511}$. We let $G = \{ g^\alpha : \alpha \in Z_q \}$ denote the subgroup generated by g. Note it has prime order, and that the exponents are from a field, namely Z_q.
>
> The secret key of a user is a random integer x in the range $\{0, \ldots, q-1\}$, and the corresponding public key is $y = g^x \bmod p$. DSA (the Digital Signature Algorithm that underlies the standard) can be used to sign any message $m \in Z_q$, as follows. The signer generates a random number $k \in \{1, \ldots, q-1\}$, which we call the *nonce*. It then computes the values $\lambda = g^k \bmod p$ and $r = \lambda \bmod q$. It sets $s = (xr + m) \cdot k^{-1} \bmod q$, where k^{-1} is the multiplicative inverse of k in the group Z_q^*. The signature of message m is the pair (r, s) and will be denoted by $\mathrm{DSA}(x, k, m)$. Note that a new, random nonce is chosen for each signature.
>
> A purported signature (r, s) of message m can be verified, given the user's public key y, by computing the values $u_1 = m \cdot s^{-1} \bmod q$, $u_2 = r \cdot s^{-1} \bmod q$ and checking that $(g^{u_1} y^{u_2} \bmod p) \bmod q = r$. Notice that the values (r, s) output by $\mathrm{DSA}(x, k, m)$ satisfy the relation $sk - rx = m \pmod q$. We will make use of this relation in our attack on the DSS.

Bellare at p. 281(highlighting added).

217. A POSA would know that $\mathbf{Z}_q$, i.e., the "range {0, …, q-1}," referenced above is a group. Also, Bellare identifies $\mathbf{Z}_q$ as a "field," which implies it is also a group (in fact, a field contains two groups, as I explain above in ¶ 68). Bellare also teaches that the exponentiation operations in DSA, as well as operations in $\mathbf{Z}_q$, are performed over a group of order q. *See, e.g.,* Bellare at pp. 281-82.

218. In addition, to the extent that the preambles to claims 15 and 23 require a "a computer-readable medium comprising instructions" and "cryptographic unit having access to computer readable instructions," Bellare in view of LEDA discloses these additional features.

219. A POSA would understand that it would be obvious to implement, and that there would an expectation of success from implementing, "the scheme" disclosed in Bellare and the C++ source code (computer executable instructions) disclosed in LEDA by storing computer executable instructions in a computer readable medium or a cryptographic unit for performing the instructions. Bellare at p. 281; LEDA at p. 93. For example, a POSA would find it obvious that the algorithms disclosed in Bellare and the C++ code disclosed in LEDA would both be implemented by storing executable instructions in a computer readable medium or a cryptographic unit, which are then performed by a processor reading those instructions from the computer readable medium. Moreover, Bellare teaches: "In some cases, the random numbers chosen may have to be kept secret (as for DSS, where the leakage of one such random number compromises." Bellare at p. 277. Bellare also teaches that other parameters have to be kept secret. *See, e.g.,* Bellare at pp. 281-83. A POSA would therefore find it obvious to implement, and would have an expectation of success from implementing, "the scheme" of the combination

of Bellare with LEDA using a cryptographic unit, which, e.g., performs the cryptographic functions taught in Bellare and has technology for protecting secret values. Further, Bellare teaches that "the scheme" it discloses is for implementing the DSA cryptographic function, such as digital signatures and generating public/private keys. Thus, Bellare in view of LEDA discloses elements 1[a], 15[a],and 23[a], all of which require "f generating a key k for use in a cryptographic function performed over a group of order q."

a. **23[b]: "an arithmetic processor"**

220. Bellare in view of LEDA discloses and renders obvious "an arithmetic processor," as required by claim 23. In particular, Bellare discloses this limitation. For example, Bellare discloses that to perform "the scheme" it discloses for DSA the method must compute an exponentiation mod p and a reduction mod q. It must also compute a multiplicative inverse in $\mathbf{Z}^*_q$. A POSA would understand that each of these operations requires an arithmetic processor, e.g., to perform the addition, multiplication, and modulo operations required for exponentiation and reduction mode q. Thus, the element "an arithmetic processor" is rendered obvious by Bellare in light of LEDA.

b. **1[b], 15[b], 23[c]: "generating a seed value SV from a random number generator;"**

221. Belleare in view of LEDA discloses and renders obvious the requirement in each of claims 1, 15, and 23 for "generating a seed value SV from a random number generator." For example, Bellare discloses that the random nonce k in "the scheme" for DSA (excerpted above) is generated when the "signer generates a random number k [in] {1, …, q-1}." Bellare at p. 281. Further, Bellare discloses that such a nonce k is, in practice, generated by a random number generator—namely, a type of random number generator known as a "pseudo-random number generator"—as follows:

2.2 Pseudo-Random Number Generators

Each time DSA is used to digitally sign a message m, a nonce k is needed. Ideally k should be a truly random number. In practice the nonces k are pseudo-random numbers produced by a pseudo-random number generator.

A pseudo-random number generator is a program $\mathcal{G}$ that on input a *seed* σ, generates a seemingly random sequence of numbers $\mathcal{G}(\sigma) = k_1, k_2, \ldots$.

Bellare at p. 282 (highlighting added).

222. Further, Bellare teaches an embodiment of a random number generator that Bellare calls a "truncated linear congruential generator," which takes an initial seed σ_0 and produces a sequence of pseudo-random numbers, σ_1, σ_2, …, according to a linear recurrence:

TRUNCATED LINEAR CONGRUENTIAL GENERATORS. For security reasons it has been suggested that only some of the bits of the number produced by a linear congruential generator be used by applications, in our case the DSA algorithm. A truncated linear congruential generator does exactly this. Let's look at this more closely. A truncated linear congruential generator is parameterized by a modulus M, two numbers $a, b \in Z_M$ and two indices l, h such that $0 \leq l \leq h \leq \lg M$. The seed is a number $\sigma_0 \in Z_M$ and the generator produces numbers in the range $\{0, \ldots, 2^{h-l} - 1\}$. The generator computes a sequence σ_i according to the linear recurrence $\sigma_{i+1} = a\sigma_i + b \bmod M$. Then, each number σ_i is truncated by taking only bits $l, \ldots, h-1$ of the number, to get the number $k_i = ((\sigma_i - (\sigma_i \bmod 2^l)) \bmod 2^h)/2^l$ which is output by the generator.

Bellare at p. 282.

223. A POSA would understand these disclosures to mean that each value σ_1, σ_2, …, is a ***seed*** generated by a random number generator, because each such value is produced by the truncated linear congruential generator and is used as a seed input to the next iteration in the truncated linear congruential generator. This understanding is further supported by the notation used in Bellare, in that each seed is identified as σ_i, for i=0,1,2,…, and Bellare says that σ denotes a "seed," as shown above in Bellare at p. 282. Further, this same analysis applies to other random number generators that use hash functions, as disclosed in Bellare, including Bellare's teachings on the use of the SHA-1 hash function. *See, e.g.,* Bellare at pp. 278, 282. Thus, Bellare discloses the requirements of elements 1[b], 15[b] and 23[c], "generating a seed value SV from a random number generator."

224. In addition, LEDA discloses a random number generator, which starts from a seed, applies a hash function to that seed, so that each subsequent value is a seed value generated by a random number generator. Further, LEDA teaches discarding the first 320 such seeds; hence, each random number that it outputs is produced as a hash function applied to a seed. Moreover, each value of the random number generator that is output is a seed (for the next iteration) that is generated by a random number generator:

> 3.5 **Random Sources**
>
> We frequently need random values in our programs. A *random source* provides an unbounded stream of integers in some range [*low* .. *high*], where *high* and *low* are *ints* with $low \leq high$ and $high - low < 2^{31}$. The size restriction comes from the fact that the implementation of random sources uses *long*s. The definition
>
> ```
> random_source S(7,319);
> ```
>
> defines a random source *S* and sets its range to [7 .. 319]. Ranges of the form $[0 .. 2^p]$ are particularly useful. Therefore we have also the definition
>
> ```
> random_source S(p);
> ```

LEDA at p. 79.

225. LEDA teaches using one hash function to generate the first 31 values and then a different hash function to generate the additional values after that:

Implementation of random_source: Our implementation of random sources follows the description in [Knu81, Vol2, section 3.2.2]. We first give the mathematics and then the program. Internally, we always generate a sequence of integers in the range $[0..2^{31}-1]$. We define 32 unsigned longs $X_0, X_1, \ldots, X_{31}$ by

Hash functions

$$X_0 = seed$$

and

$$X_i = (1103515245 \cdot X_{i-1} + 12345) \bmod m$$

for $1 \leq i \leq 31$. Here $m = 2^{32}$. We extend this sequence by

$$X_i = (X_{i-3} + X_{i-32}) \bmod m$$

for $i \geq 32$. In this way an infinite sequence $X_0, X_1, \ldots$ of unsigned longs is obtained. Following [Knu81, Vol2, section 3.2.2], we discard the first 320 elements of this sequence (they are considered as a warm-up phase of the generator) and we also drop the right-most bit of each number (since it is the least random). Thus, the i-th number output by the internal generator is

```
(X[i + 320] >> 1) & 0x7fffffff.
```

LEDA at p. 92 (annotations in red and highlighting in yellow added). Further, based on these teachings, a POSA would find it obvious to combine Bellare and LEDA to give either embodiment mentioned above for the combination. That is, combining Bellare with LEDA leads to an expectation of success that the combination can produce seed values from either a hash-based random number generator like that in LEDA or a SHA-1 RNG like that in Bellare, which advantageously doubles the user's options for generating the seed value. That is, the Bellare-LEDA embodiment with the truncated linear congruential generator or the Bellare-LEDA embodiment with SHA-1 each provide benefits that a POSA would readily

appreciate, and which a POSA would have an expectation of success would result from the combination of Bellare in view of LEDA.

226. Thus, claim elements 1[b], 15[b], and 23[c], "generating a seed value SV from a random number generator" are rendered obvious by Bellare in view of LEDA.

c. **1[c], 15[c], 23[d]: "performing a hash function H( ) on said seed value SV to provide an output H(SV);"**

227. Bellare in view of LEDA discloses and renders obvious the requirement in each of claims 1, 15, and 23 for "performing a hash function H( ) on said seed value SV to provide an output H(SV)." For example, as noted above for elements 1[b], 15[b], and 23[c], Bellare discloses that a truncated linear congruential generator can be used as a random number generator. A POSA would understand from this disclosure, for instance, that each value σ_{i+1} is computed by performing a hash function on a seed value, σ_i, consisting of the previous value generated (or σ_0 in the case of the first random number generated), as a part of a truncated linear congruential generator. Bellare at p. 282. A POSA would understand that the hash function disclosed in this case is a multiplicative hash function, $\sigma_{i+1} = a*\sigma_i + b \pmod M$, where a, b, and M are parameters of the random number generator. That is, this passage in Bellare discloses applying a hash function, $H(x) = a*x+ b \pmod M$, to said seed value from elements 1[b], 15[b] and 23[c]. *Id.* Bellare also teaches the

use of "the use of a pseudo-random generator based on SHA-1 or DES." Bellare at p. 278. A POSA would understand that SHA-1 is a hash function, and that this teaching in Bellare discloses it should be applied as hash function in the random number generator for generating the nonce k. Thus, Bellare discloses elements 1[c], 15[c] and 23[d], "performing a hash function H( ) on said seed value SV to provide an output H(SV)."

228. Likewise, LEDA also teaches the use of a hash function to generate random numbers, where each hash function is applied to a seed (after an initial seed X_0) that is generated by a random number generator:

Implementation of random_source: Our implementation of random sources follows the description in [Knu81, Vol2, section 3.2.2]. We first give the mathematics and then the program. Internally, we always generate a sequence of integers in the range $[0\,..\,2^{31}-1]$. We define 32 unsigned longs $X_0, X_1, \ldots, X_{31}$ by

Hash functions

$$X_0 = seed$$

and

$$X_i = (1103515245 \cdot X_{i-1} + 12345) \bmod m$$

for $1 \leq i \leq 31$. Here $m = 2^{32}$. We extend this sequence by

$$X_i = (X_{i-3} + X_{i-32}) \bmod m$$

for $i \geq 32$. In this way an infinite sequence $X_0, X_1, \ldots$ of unsigned longs is obtained. Following [Knu81, Vol2, section 3.2.2], we discard the first 320 elements of this sequence (they are considered as a warm-up phase of the generator) and we also drop the right-most bit of each number (since it is the least random). Thus, the i-th number output by the internal generator is

```
(X[i + 320] >> 1) & 0x7fffffff.
```

LEDA at p. 92 (annotations in red and highlighting in yellow added). Thus, either of the combinations Bellare-LEDA with a truncated linear congruential generator or Bellare-LEDA with SHA-1 renders claim elements 1[c], 15[c] and 23[c] obvious. That is, combining Bellare with LEDA thus leads to an expectation of success that the combination can produce seed values from either a hash-based random number generator like that in LEDA or a SHA-1 RNG like that in Bellare, which advantageously doubles the user's options for generating the seed value.

229. Therefore, claim elements 1[c], 15[c], and 23[d], "performing a hash function H( ) on said seed value SV to provide an output H(SV);" is rendered obvious by Bellare in view of LEDA.

d. **1[d], 25[d], 23[e]: "determining whether said output H(SV) is less than said order q prior to reducing mod q;"**

230. Claim elements 1[d], 15[d], and 23[e], "determining whether said output H(SV) is less than said order q prior to reducing mod q" are rendered obvious by Bellare in view of LEDA. Bellare discloses that in "the scheme," the signer generates a random number k in the range from 1 to q-1, called a *nonce*:

THE SCHEME. The scheme uses the following parameters: a prime number p, a prime number q which divides $p-1$ and an element $g \in Z_p^*$ of order q. (Chosen as $g = h^{(p-1)/q}$ where h is a generator of the cyclic group Z_p^*). These parameters may be common to all users of the signature scheme and we will consider them as fixed in the rest of the paper. The standard asks that $2^{159} < q < 2^{160}$ and $p > 2^{511}$. We let $G = \{ g^\alpha : \alpha \in Z_q \}$ denote the subgroup generated by g. Note it has prime order, and that the exponents are from a field, namely Z_q.

The secret key of a user is a random integer x in the range $\{0, \ldots, q-1\}$, and the corresponding public key is $y = g^x \bmod p$. DSA (the Digital Signature Algorithm that underlies the standard) can be used to sign any message $m \in Z_q$, as follows. The signer generates a random number $k \in \{1, \ldots, q-1\}$, which we call the *nonce*. It then computes the values $\lambda = g^k \bmod p$ and $r = \lambda \bmod q$. It sets $s = (xr + m) \cdot k^{-1} \bmod q$, where k^{-1} is the multiplicative inverse of k in the group Z_q^*. The signature of message m is the pair (r, s) and will be denoted by $\mathrm{DSA}(x, k, m)$. Note that a new, random nonce is chosen for each signature.

Bellare at p. 281 (highlighting added).

231. In addition, Bellare teaches the use of possible random number generators based on hash functions applied to a seed value (generated from a random number generator in a previous iteration) to generate the key k.

232. Bellare does not explicitly disclose "determining whether said output H(SV) is less than said order q prior to reducing mod q." Nevertheless, Bellare does disclose that the nonce k used in "the scheme" for DSA needs to be as random as possible in order for "the scheme" to be secure. *See, e.g.,* Bellare at p. 278 ("We remark that the Standard [16] recommends the use of a pseudo-random generator based on SHA-1 or DES. The attack we describe does not say anything about the use of DSS with such generators, but it does illustrate[] the high vulnerability of the

DSS to the underlying random number generation process."). Thus, as I explain above with respect to the motivation to combine Bellare with LEDA, a POSA would look to a textbook like LEDA for methods that improve the randomness and uniformity of the nonce k.

233. For example, LEDA discloses that to get a random value in the range from low to high, simply reducing a random value modulo high-low+1 will produce a biased result:

> We next show how to generate a number uniformly at random in [*low* .. *high*]. Let *X* be a number produced by the internal generator. Then $low+X \bmod(high-low+1)$ is a number in the range [*low* .. *high*]. However, this number is not uniformly distributed (consider the case where $low = 0$ and $high = 2^{31} - 2$ and observe that in this case the number 0 is generated with probability twice as large as any other number). We therefore proceed differently. Our

LEDA at p. 92 (highlighting added).

234. LEDA then explains how to apply rejection sampling to a value, x, produced by a hash function, internal_source(), which is based on using a seed generated by a random number generator of type random_source, as explained above. *See, e.g.,* LEDA at p. 92. LEDA discloses a random source based on a linear congruential generator that uses a hash function that takes as an input seed the output random value from the previous iteration, to generate a random number:

Implementation of random_source: Our implementation of random sources follows the description in [Knu81, Vol2, section 3.2.2]. We first give the mathematics and then the program. Internally, we always generate a sequence of integers in the range $[0 .. 2^{31} - 1]$. We define 32 unsigned longs $X_0, X_1, \ldots, X_{31}$ by

Hash functions

$$X_0 = seed$$

and

$$X_i = (1103515245 \cdot X_{i-1} + 12345) \bmod m$$

for $1 \leq i \leq 31$. Here $m = 2^{32}$. We extend this sequence by

$$X_i = (X_{i-3} + X_{i-32}) \bmod m$$

for $i \geq 32$. In this way an infinite sequence $X_0, X_1, \ldots$ of unsigned longs is obtained. Following [Knu81, Vol2, section 3.2.2], we discard the first 320 elements of this sequence (they are considered as a warm-up phase of the generator) and we also drop the right-most bit of each number (since it is the least random). Thus, the i-th number output by the internal generator is

```
(X[i + 320] >> 1) & 0x7fffffff.
```

LEDA at p. 92 (annotations in red and highlighting in yellow added).

235. LEDA then discloses how to use this hash value to generate a random number uniformly distributed in a given range, [low..high]. LEDA computes a difference, diff=high-low, which in the case of the range [0,q-1] implies that diff=q-1. If x is greater than diff (that is, greater than or equal to q in this case), then LEDA teaches repeating the hash-function process (in a while loop). That is, rather than reducing mod q, which LEDA teaches would be biased, LEDA teaches rejecting a value of x that is too large and repeating the hash function computation. Thus, the comparison "x > diff" in the following excerpt from LEDA discloses "determining whether said output H(SV) is less than said order q prior to reducing mod q":

```
⟨generation of a random number in [low..high]⟩≡
  int diff = high - low;
  /* compute pat = 2^p - 1 with 2^{p-1} <= diff < 2^p */
  unsigned long pat = 1;
  while (pat <= diff) pat <<= 1;
  pat--;
  /* pat = 0...01...1 with exactly p ones.
     Now, generate random x in [0 .. pat]
     until x <= diff and return low + x    */
  unsigned long x = internal_source() & pat;
  while ( x > diff) x = internal_source() & pat;
  return (int)(low + x);
```

LEDA at p. 93 (highlighting added).

236. As mentioned above, the "& pat" expression is a bit-wise AND of the value returned by the hash function "internal_source()," which is the LEDA implementation of "random_source." This would be understood by a POSA as a hash function (i.e., combining the hash function "internal_source()" with a hash function that is restricted to the p least-significant bits). Thus, for the remainder of this analysis, I refer to the hash function "internal_source()" to be inclusive of this bit-wise ***and*** operation.

237. Further, it would be obvious to substitute this method from LEDA (of applying rejection sampling with a hash value applied to a seed produced by a random number generator) for the method of Bellare (of using a hash value applied to a seed produced by a random number generator and reducing it modulo q), since there would be an expectation of success to a POSA that substituting one module

(reducing mod q) for another (using a "while" loop based on rejection sampling that terminates when x is not greater than diff=q-1, that is, it terminates when x is less than q) would reduce bias via a well-known rejection sampling technique. Further, since LEDA teaches that reducing modulo q (which is equal to high-low+1 in this case) is "not uniformly distributed," it would be obvious to a POSA that there would be a further expectation of success from this substitution to reduce bias, since by having the nonce k be uniformly distributed in the range [0,q-1] it won't be biased:

Implementation of random_source: Our implementation of random sources follows the description in [Knu81, Vol2, section 3.2.2]. We first give the mathematics and then the program. Internally, we always generate a sequence of integers in the range $[0\,..\,2^{31}-1]$. We define 32 unsigned longs $X_0, X_1, \ldots, X_{31}$ by

$$X_0 = seed$$

and

$$X_i = (1103515245 \cdot X_{i-1} + 12345) \bmod m$$

for $1 \leq i \leq 31$. Here $m = 2^{32}$. We extend this sequence by

$$X_i = (X_{i-3} + X_{i-32}) \bmod m$$

for $i \geq 32$. In this way an infinite sequence $X_0, X_1, \ldots$ of unsigned longs is obtained. Following [Knu81, Vol2, section 3.2.2], we discard the first 320 elements of this sequence (they are considered as a warm-up phase of the generator) and we also drop the right-most bit of each number (since it is the least random). Thus, the i-th number output by the internal generator is

```
(X[i + 320] >> 1) & 0x7fffffff.
```

We next show how to generate a number uniformly at random in $[low\,..\,high]$. Let X be a number produced by the internal generator. Then $low + X \bmod (high - low + 1)$ is a number in the range $[low\,..\,high]$. However, this number is not uniformly distributed (consider the case where $low = 0$ and $high = 2^{31} - 2$ and observe that in this case the number 0 is generated with probability twice as large as any other number). We therefore proceed differently. Our

LEDA at p. 92 (highlighting added). Further, a POSA would understand that this same analysis applies to either of the above-mentioned embodiments of a combination of Bellare and LEDA, e.g., where the hash function used in LEDA is replaced with the truncated linear congruential generator disclosed in Bellare (*see, e.g.,* Bellare at p. 282) or with SHA-1, as Bellare recommends. *See, e.g.,* Bellare at p. 278. That is, combining Bellare with LEDA thus leads to an expectation of success that the combination can produce seed values from either a hash-based random number generator like that in LEDA or a SHA-1 RNG like that in Bellare, which advantageously doubles the user's options for generating the seed value while also reducing bias.

238. Thus, claim elements 1[d], 15[d], and 23[e], "determining whether said output H(SV) is less than said order q prior to reducing mod q;" are rendered obvious by Bellare in view of LEDA.

e. **1[e], 15[e], 23[f]: "accepting said output H(SV) for use as said key k if the value of said output is less than said order q;"**

239. Claim elements 1[e], 15[e], and 23[f], "accepting said output H(SV) for use as said key k if the value of said output is less than said order q" are rendered obvious by Bellare in view of LEDA. For example, as I explain for claim elements 1[d], 15[d], and 23[e] above, Bellare in view of LEDA teaches comparing the output

of the hash function (applied to the seed) to q in a "while" loop that terminates when the value x is less than q, the order of the group disclosed in Bellare. The method in LEDA then returns low+x, which is simply x in the case when low=0, i.e., for the interval [0,q-1]:

⟨generation of a random number in [low..high]⟩≡

```
int diff = high - low;
/* compute pat = 2^p - 1 with 2^{p-1} <= diff < 2^p */
unsigned long pat = 1;
while (pat <= diff) pat <<= 1;
pat--;
/* pat = 0...01...1 with exactly p ones.
   Now, generate random x in [0 .. pat]
   until x <= diff and return low + x    */
unsigned long x = internal_source() & pat;
while ( x > diff) x = internal_source() & pat;
return (int)(low + x);
```

LEDA at p. 93 (highlighting added). Because the "while" loop terminates when the value x is less than q, this would indicate that the output is accepted for use as a key, as required by the claim. Further, this random-number generation method applies equally to the embodiment of Bellare-LEDA using the truncated linear congruential generator of Bellare or with the SHA-1 cryptographic hash function recommended by Bellare. That is, combining Bellare with LEDA thus leads to an expectation of success that the combination can produce seed values from either a hash-based random number generator like that in LEDA or a SHA-1 RNG like that in Bellare,

which advantageously doubles the user's options for generating the seed value while also reducing bias.

240. Thus, claim elements 1[e], 15[e], and 23[f], "accepting said output H(SV) for use as said key k if the value of said output is less than said order q" are rendered obvious by Bellare in view of LEDA.

f. **1[f], 15[f], 23[g], "rejecting said output H(SV) as said key if said value is not less than said order q;"**

241. Claim elements 1f], 15[f], and 23[g], "rejecting said output H(SV) as said key if said value is not less than said order q" are rendered obvious by Bellare in view of LEDA. For example, as I explain for claim elements 1[d], 15[d], 23[e], and 1[e], 15[e], 23[f] above, Bellare in view of LEDA teaches comparing the output of the hash function (applied to the seed) to q in a "while" loop that terminates when the value x is less than q, the order of the group disclosed in Bellare. When the value x is greater than q, the loop continues. Thus, the hash value returned by "internal_source()" is rejected if the value x is greater than diff, i.e., when x is not less than q (since diff=q-1 when applying the teachings of LEDA for the interval [0,q-1]). Further, this random-number generation method using a comparison with q applies equally to the embodiment of Bellare-LEDA using the truncated linear congruential generator of Bellare or with the SHA-1 cryptographic hash function recommended by Bellare. That is, combining Bellare with LEDA thus leads to an

expectation of success that the combination can produce seed values from either a hash-based random number generator like that in LEDA or a SHA-1 RNG like that in Bellare, which advantageously doubles the user's options for generating the seed value while also reducing bias.

242. Thus, claim elements 1[f], 15[f], and 23[g], "rejecting said output H(SV) as said key if said value is not less than said order q," are rendered obvious by Bellare in view of LEDA.

g. 1[g], 15[g], 23[h]: "if said output H(SV) is rejected, repeating said method; and"

243. Claim elements 1[g], 15[g], and 23[h], "if said output H(SV) is rejected, repeating said method" are rendered obvious by Bellare in view of LEDA. For example, as I explain with respect to the three prior steps of the independent claims, Bellare in view of LEDA teaches comparing the output of the hash function (applied to the seed) to q in a "while" loop that terminates when the value x is less than q, the order of the group disclosed in Bellare. The loop continues until the value of x is greater than q. Thus, the hash value returned by "internal_source()" is rejected if the value x is greater than diff, i.e., when x is not less than q (since diff=q-1 when applying the teachings of LEDA for the interval [0,q-1]), and the method is repeated, calling internal_source() again, if the output is rejected. Further, this random-number generation method using rejection sampling applies equally to the

embodiment of Bellare-LEDA using the truncated linear congruential generator of Bellare or with the SHA-1 cryptographic hash function recommended by Bellare. That is, combining Bellare with LEDA leads to an expectation of success that the combination can produce seed values from either a hash-based random number generator like that in LEDA or a SHA-1 RNG like that in Bellare, which advantageously doubles the user's options for generating the seed value while also reducing bias.

244. Thus, claim elements 1[g], 15[g], and 23[h], "if said output H(SV) is rejected, repeating said method," are rendered obvious by Bellare in view of LEDA.

h. **1[h], 15[h], and 23[i]: "if said output H(SV) is accepted, providing said key k for use in performing said cryptographic function, wherein said key k is equal to said output H(SV)."**

245. Claim elements 1[h], 15[h], and 23[i], "if said output H(SV) is accepted, providing said key k for use in performing said cryptographic function, wherein said key k is equal to said output H(SV)," are rendered obvious by Bellare in view of LEDA. For example, as I explain with respect to the earlier steps of claims 1, 15, and 23 above, Bellare in view of LEDA teaches comparing the output of the hash function (applied to the seed) to q in a while loop that terminates when the value x is less than q, the order of the group disclosed in Bellare. Further, the value low+x (which is equal to x when low=0) is returned by the method of LEDA

when the while-loop terminates, i.e., when the hash value returned by "internal_source()" is accepted. LEDA at p. 93. Since this value would then be used in "the scheme" by Bellare, according to the substitution of the method of LEDA for the reduction modulo q taught in Bellare, there would be an expectation of success to a POSA that the returned value, x, would then be used as the key k in "the scheme" of Bellare. As mentioned above, this is a cryptographic function. Further, this random-number generation method applies equally to the embodiment of Bellare-LEDA using the truncated linear congruential generator of Bellare or with the SHA-1 cryptographic hash function recommended by Bellare. That is, as already noted, this random-number generation method using rejection sampling applies equally to the embodiment of Bellare-LEDA using the truncated linear congruential generator of Bellare or with the SHA-1 cryptographic hash function recommended by Bellare. That is, combining Bellare with LEDA leads to an expectation of success that the combination can produce seed values from either a hash-based random number generator like that in LEDA or a SHA-1 RNG like that in Bellare, which advantageously doubles the user's options for generating the seed value while also reducing bias.

246. Thus, claim elements 1[h], 15[h], and 23[i], "if said output H(SV) is accepted, providing said key k for use in performing said cryptographic function,

wherein said key k is equal to said output H(SV)," are rendered obvious by Bellare in view of LEDA.

247. Therefore, claims 1, 15, and 23 of the '961 patent are rendered obvious by the combination of Bellare in view of LEDA.

ii. Claims 2, 16, and 24

248. Claims 2, 16, and 24 are all dependent claims that follows from independent claims 1, 15, and 23, respectively, and require: "wherein another seed value is generated by said random number generator if said output is rejected." In my opinion, claims 2, 16, and 24 of the '961 patent are rendered obvious by Bellare in view of LEDA. I incorporate by reference my analysis for claims 1, 15, and 23.

249. With respect to the additional limitation in claims 2, 16, and 24, in the "while" loop disclosed in LEDA, the random number generator, "internal_source()", is called again when the previous value is rejected. LEDA at p. 93. According to the truncated linear congruential generator taught in Bellare, the hash function taught in LEDA, as well as using SHA-1 as taught in Bellare, the next call to "internal_source()" is based on using the previous generated value now as a seed. Bellare at pp. 278, 282; LEDA at p. 92. That is, the next seed to "internal_source()" is generated by said random number generator if the output is rejected. Thus, the requirement in claims 2, 16, and 24 "wherein another seed value is generated by said

random number generator if said output is rejected" is rendered obvious by Bellare in view of LEDA.

iii. Claims 3, 17, and 25

250. Claims 3, 17, and 25 are dependent claims that follow from the independent claims, and require: "wherein the step of accepting said output as a key includes a further step of storing said key." In my opinion, claims 3, 17, and 25 of the '961 patent are rendered obvious by Bellare in view of LEDA. I incorporate by reference my analysis for claims 1, 15, and 23.

251. With respect to the additional limitations in claim 3, 17, and 25, LEDA teaches storing the value returned from "internal_source()" in a variable, x, and returning this value (when low=0, as in the case for an interval, [0,q-1]), to the calling function, which in this combination is "the scheme" of Bellare. Bellare at p. 281; LEDA at p. 93. This returned value is then stored in the variable k according to "the scheme" of Bellare. *See, e.g.,* Bellare at p. 281. Thus, claims 3, 17, and 25 "wherein the step of accepting said output as a key includes a further step of storing said key" are rendered obvious by Bellare in view of LEDA.

iv. Claims 4, 18, and 26

252. Claims 4, 18, and 26 are dependent claims that follow from the independent claims, and require: " wherein said key is used for generation of a public

key." In my opinion, claims 4, 18, and 26 of the '961 patent are rendered obvious by Bellare in view of LEDA. I incorporate by reference my analysis for claims 1, 15, and 23.

253. In addition, Bellare discloses that the key is used for generating a public key. It discloses that function in "the scheme" of DSA for signing a sequence of message, m_i, each key, r_i, which is used by user B to decrypt the signature, s_i:

> The adversary (cryptanalyst) sees the public key y, and triples (m_i, r_i, s_i) where (r_i, s_i) is a signature of m_i. Notice that the secrecy of the nonces is crucial. If ever a single nonce k_i is revealed to the adversary, then the latter can recover the secret key x, because $x = (s_i k_i - m_i) r_i^{-1} \bmod q$. However, the nonces appear to be very well protected, making it hard to exploit any such weakness. The
>
> cryptanalyst only sees $r_i = (g^{k_i} \bmod p) \bmod q$ from which he cannot recover k_i short of computing discrete logarithms, and in fact not even then, due to the second mod operation. So even if $\mathcal{G}$ is a predictable generator, meaning, say, that given k_1, k_2 we can find k_3, there is no a priori reason to think DSS is vulnerable with this generator, because how can the cryptanalyst ever get to know k_1, k_2 anyway?

Bellare at pp. 278-279 (highlighting added).

254. Here, the referenced "cryptanalyst" can see each key r_i. Thus, a POSA would recognize that each key r_i is a (short-term ephemeral) public key (e.g., it is sent to the signature verifier to decrypt a digital signature) and that it has an associated private key, k_i; hence, a POSA would understand each r_i to be a public key.

255. Alternatively, Bellare teaches that the public key, y, used in DSS is generated from a random secret key, x, which is in $\mathbf{Z}_q$, i.e., x is a secret key chosen randomly in the range from 1 to q-1:

1.1 Pseudorandom numbers in DSS

Recall that the DSS has public parameters p, q, g where p, q are primes, of 512 bits and 160 bits respectively, and g is a generator of an order q subgroup of Z_p^*. The signer has a public key $y = g^x$ where $x \in Z_q$. To sign a message $m \in Z_q$, the signer picks at random a number $k \in \{1, \ldots, q-1\}$ and computes a signature (r, s), where $r = (g^k \bmod p) \bmod q$ and $s = (xr + m)k^{-1} \bmod q$.

Here the "nonce" k is chosen at random, anew for each message. In practice, a sequence of nonces will be produced by a generator $\mathcal{G}$ which, given some initial seed k_0, produces a sequence of values $k_1, k_2, \ldots$; k_i will be the nonce for the i-th signature.

The adversary (cryptanalyst) sees the public key y, and triples (m_i, r_i, s_i) where (r_i, s_i) is a signature of m_i. Notice that the secrecy of the nonces is crucial. If ever a single nonce k_i is revealed to the adversary, then the latter can recover the secret key x, because $x = (s_i k_i - m_i) r_i^{-1} \bmod q$. However, the nonces appear to be very well protected, making it hard to exploit any such weakness. The

Bellare at p. 278 (highlighting added).

256. Thus, a POSA would find it obvious to use the same method to generate x as would be used to generate the random nonce, k, disclosed in "the scheme" of DSA. Thus, for the same reasons given above for why the method of claims 1, 15, and 23 of the '961 patent for generating k are obvious, a similar method for generating the secret key, x, which is then used to generate the public key, y, also is rendered obvious.

257. Thus, claims 4, 18, and 26 "wherein said key is used for generation of a public key," are rendered obvious by Bellare in view of LEDA.

v. Claim 5, 19, and 27

258. Claims 5, 19, and 27 are dependent claims that depend from the independent claims, and require: "wherein said order q is a prime number represented by a bit string of predetermined length L." In my opinion, claims 5, 19, and 27 of the '961 patent are rendered obvious by Bellare in view of LEDA. I incorporate by reference my analysis for claims 1, 15, and 23.

259. With respect to the additional limitation in claims 5, 19, and 27, LEDA teaches storing the value returned from "internal_source()" in a variable, x, and returning this value (when low=0, as in the case for an interval, [0,q-1]), to the calling function, which in this combination is "the scheme" of Bellare. Bellare at p. 281; LEDA at p. 93. This returned value is then stored in the variable k according to "the scheme" of Bellare. Bellare at p. 281. Further, "the scheme" disclosed in Bellare teaches that the order q is a prime number represented by a bit string of predetermined length 160. *See, e.g.,* Bellare at pp. 278, 281. Further, this predetermined length of 160 bits applies equally to the embodiment of Bellare-LEDA using the truncated linear congruential generator of Bellare or with the SHA-1 cryptographic hash function recommended by Bellare, which is advantageous

since it reduces bias. Thus, claims 5, 19, and 27 "wherein said order q is a prime number represented by a bit string of predetermined length L" are rendered obvious by Bellare in view of LEDA.

vi. Claims 6, 20, and 28

260. Claims 6, 20, and 28 are multiply dependent claims that follow from the dependent claims 5, 19, and 27, and require: "wherein said output from said hash function is a bit string of predetermined length L." In my opinion, claims 6, 20, and 28 of the '961 patent are rendered obvious by Bellare in view of LEDA. For example, I incorporate by reference my analysis above for claims 5, 19, and 27.

261. With respect to the additional limitation of claims 6, 20, and 28, Bellare teaches that its hash function used in conjunction with its truncated linear congruential generator produces outputs represented by a bit string of predetermined length *h*-*l*, where *h* and *l* are input indices, and that the resulting nonce k has 160 bits. Bellare at p. 282. In addition, Bellare discloses that the SHA-1 hash function recommended by Bellare for computing the key k in DSA produces an output that is a bit string of predetermined length 160 bits. Bellare at pp. 278, 281. Further, this use of a 160-bit hash function applies equally to the embodiment of Bellare-LEDA using the truncated linear congruential generator of Bellare or with the SHA-1 cryptographic hash function recommended by Bellare, which is advantageous

because it reduces bias for two different methods. Thus, in either embodiment of a combination of Bellare and LEDA, a POSA would find it obvious for the output of the hash function to be a bit string of predetermined length L. Thus, in my opinion, claims 6, 20, and 28 of the '961 patent are rendered obvious by Bellare in view of LEDA.

E. Motivation to combine

262. A POSA would be motivated combine the teachings of Bellare with the teachings of LEDA, at least because a POSA would be motivated to improve the randomness and uniformity of the seed of Bellare using techniques from LEDA, as applied to the cryptographic methods in Bellare. This amounts to a substitution of known techniques, and moreover, such a combination would have a reasonable expectation of success.

263. A POSA would understand the disclosures in Bellare to teach the steps of the DSA algorithm (in Bellare, referred to as the DSS algorithm), as disclosed in "the scheme" algorithm description in Bellare:

THE SCHEME. The scheme uses the following parameters: a prime number p, a prime number q which divides $p-1$ and an element $g \in Z_p^*$ of order q. (Chosen as $g = h^{(p-1)/q}$ where h is a generator of the cyclic group Z_p^*). These parameters may be common to all users of the signature scheme and we will consider them as fixed in the rest of the paper. The standard asks that $2^{159} < q < 2^{160}$ and $p > 2^{511}$. We let $G = \{ g^\alpha : \alpha \in Z_q \}$ denote the subgroup generated by g. Note it has prime order, and that the exponents are from a field, namely Z_q.

The secret key of a user is a random integer x in the range $\{0, \ldots, q-1\}$, and the corresponding public key is $y = g^x \bmod p$. DSA (the Digital Signature Algorithm that underlies the standard) can be used to sign any message $m \in Z_q$, as follows. The signer generates a random number $k \in \{1, \ldots, q-1\}$, which we call the *nonce*. It then computes the values $\lambda = g^k \bmod p$ and $r = \lambda \bmod q$. It sets $s = (xr + m) \cdot k^{-1} \bmod q$, where k^{-1} is the multiplicative inverse of k in the group Z_q^*. The signature of message m is the pair (r, s) and will be denoted by $\mathrm{DSA}(x, k, m)$. Note that a new, random nonce is chosen for each signature.

A purported signature (r, s) of message m can be verified, given the user's public key y, by computing the values $u_1 = m \cdot s^{-1} \bmod q$, $u_2 = r \cdot s^{-1} \bmod q$ and checking that $(g^{u_1} y^{u_2} \bmod p) \bmod q = r$. Notice that the values (r, s) output by $\mathrm{DSA}(x, k, m)$ satisfy the relation $sk - rx = m \pmod q$. We will make use of this relation in our attack on the DSS.

Bellare at p. 281 (highlighting added).

264. In addition, a POSA would understand that Bellare teaches that the random number used for the nonce k in the DSA "the scheme" algorithm should be uniformly chosen in the interval from 1 to q-1 as randomly as possible:

> The intent of our paper is to illustrate the extreme care with which one should choose a pseudo random number generator to use within a particular cryptographic algorithm. Specifically, we consider a concrete algorithm, the Digital Signature Standard [16], and a concrete pseudo random number generator, the linear congruential generator (LCG) or truncated linear congruential pseudo random generator. We then show that if a LCG or truncated LCG is used to produce the pseudo random choices called for in DSS, then DSS becomes completely breakable.
>
> We remark that the Standard [16] recommends the use of a pseudo-random generator based on SHA-1 or DES. The attack we describe does not say anything about the use of DSS with such generators, but it does illustrates the high vulnerability of the DSS to the underlying random number generation process.

Bellare at p. 278 (emphasis added).

265. Bellare likewise stresses that the security of DSA depends on the secrecy and randomness of the nonce, k. For example, Bellare discloses the following about the secrecy of the nonce:

> SECRECY OF THE NONCE. Recall that for every signature, the signer generates a new, random nonce k. An important feature (drawback!) of the DSS is that the security relies on the secrecy of the nonces. If any nonce k ever becomes revealed, at any time, even long after the signature (r, s) was generated, then given the nonce and the signature one can immediately recover the secret key x, via $x = (sk - m)r^{-1} \bmod q$. This is a key point in our attack.

Bellare at p. 281.

266. Thus, a POSA reading Bellare would be motivated to seek methods to improve the randomness and uniformity of the nonce, k, which Bellare discloses should be a random number uniformly chosen in the range from 1 to q-1. The LEDA reference describes how to use a random-bit generator to generate random numbers that are uniformly distributed in a given interval. *See* LEDA at pp. 92-93. Thus, a

POSA would look to LEDA for such methods. Further, LEDA teaches that reducing the function modulo (high-low+1) to generate a random number in a range [low,high] will produce a biased result. LEDA at p. 92. Thus, a POSA would understand that LEDA's method of generating a random number (based on rejection sampling) instead generates a random number that is uniformly distributed in the range and also advantageously reduces bias:

> We next show how to generate a number uniformly at random in [*low* .. *high*]. Let X be a number produced by the internal generator. Then $low+X \bmod(high-low+1)$ is a number in the range [*low* .. *high*]. However, this number is not uniformly distributed (consider the case where $low = 0$ and $high = 2^{31} - 2$ and observe that in this case the number 0 is generated with probability twice as large as any other number). We therefore proceed differently. Our

LEDA at p. 92 (highlighting added).

267. Bellare also teaches the use of hash functions with respect to the random number generator used to generate the nonce k in Bellare's description of "the scheme" for the DSS standard. For example, Bellare teaches the use of a "truncated linear congruential generator," which uses a multiplicative hash function composed with a hash function that truncates the bits of a multiplicative hash function:

TRUNCATED LINEAR CONGRUENTIAL GENERATORS. For security reasons it has been suggested that only some of the bits of the number produced by a linear congruential generator be used by applications, in our case the DSA algorithm. A truncated linear congruential generator does exactly this. Let's look at this more closely. A truncated linear congruential generator is parameterized by a modulus M, two numbers $a, b \in Z_M$ and two indices l, h such that $0 \leq l \leq h \leq \lg M$. The seed is a number $\sigma_0 \in Z_M$ and the generator produces numbers in the range $\{0, \ldots, 2^{h-l} - 1\}$. The generator computes a sequence σ_i according to the linear recurrence $\sigma_{i+1} = a\sigma_i + b \bmod M$. Then, each number σ_i is truncated by taking only bits $l, \ldots, h-1$ of the number, to get the number $k_i = ((\sigma_i - (\sigma_i \bmod 2^l)) \bmod 2^h)/2^l$ which is output by the generator.

Bellare at p. 282.

268. Bellare also teaches the use of a random generator based on SHA-1, which is the cryptographic hash function, as discussed above in ¶ 74, to generate the nonce k of "the scheme" for the DSS standard:

> We remark that the Standard [16] recommends the use of a pseudo-random generator based on SHA-1 or DES. The attack we describe does not say anything about the use of DSS with such generators, but it does illustrates the high vulnerability of the DSS to the underlying random number generation process.

Bellare at p. 278.

269. As a result, a POSA would be motivated to replace the random number generator taught in LEDA (which is implemented in the "internal_source()" function) with either the truncated linear congruential generator or the cryptographic hash function, SHA-1, and then use this method to generate the nonce k disclosed in "the scheme" method taught in Bellare, because the combination advantageously

reduces bias via LEDA's rejection sampling. For example, the source code disclosed in LEDA for generating a random number uniformly in the range from low to high is disclosed as follows:

```
⟨generation of a random number in [low..high]⟩≡
  int diff = high - low;
  /* compute pat = 2^p - 1 with 2^{p-1} <= diff < 2^p */
  unsigned long pat = 1;
  while (pat <= diff) pat <<= 1;
  pat--;
  /* pat = 0...01...1 with exactly p ones.
     Now, generate random x in [0 .. pat]
     until x <= diff and return low + x    */
  unsigned long x = internal_source() & pat;
  while ( x > diff) x = internal_source() & pat;
  return (int)(low + x);
```

LEDA at 93.

270. The above-excerpted source code disclosed in the LEDA textbook is written so that internal_source() can be implemented using any random-number generator. Because of this, a POSA would recognize that implementing internal_source() with SHA-1 or the truncated linear congruential generator of Bellare amounts to a substitution of one known element for another to obtain a predictable result, which advantageously produces an expectation of success in reducing bias. Namely, a POSA would be motivated to substitute one type of random number generator (the hash-based random number generator disclosed in LEDA, which Bellare criticizes as generating having a bias) for internal_source()

with a cryptographic hash function, such as the truncated linear congruential generator of Bellare (which uses a longer bit string of 160 bits than the 32-bit generator of LEDA) or with SHA-1 (which Bellare presents as an even better option), to achieve the expected result of uniformly generating a random number in a given range, thereby improving the randomness of the nonce k. Further, since the method taught in LEDA begins with a random seed, a POSA would also be motivated to use the same seed in this substitution. LEDA at p. 92. Thus, a POSA would have a reasonable expectation of success of improving the uniformity and randomness of the nonce k with this combination, while also reducing bias, and it would be obvious to try this combination of Bellare with LEDA.

F. Alleged Secondary Considerations of Non-Obviousness

271. In presenting my opinions, I have considered whether there might be any secondary considerations of non-obviousness. I have not observed any that would change my conclusions that the claims of the ’961 patent are obvious in view of either the combination of Miyaji in view of ordinary skill in the art, or Bellare in view of LEDA. For instance, I considered what was raised about secondary considerations in the earlier Facebook IPR2019-0923, and see nothing there that changes my conclusions about obviousness.

272. I examined excerpts of Blackberry's Supplemental Responses and Objections to Facebook's Interrogatories provided as an Exhibit in IPR2019-0923 ("IPR-Rogs"). This excerpted exhibit (Exhibit 1027 to this Petition) includes Patent Owner's response to Interrogatory No. 5, which asks Patent Owner to identify secondary considerations of non-obviousness, including identifying any factual and legal basis regarding the purported nexus between the secondary considerations and the claims at issue in that action. Ex. 1027 at p. 2.

273. As I explain below, the evidence I have examined for alleged secondary considerations of non-obviousness do not, in my opinion, overcome the obviousness opinions that I provide above. I reserve the right to supplement my opinions on this issue should Patent Owner attempt to introduce any evidence directed at establishing that there are, in fact, secondary considerations of non-obviousness.

i. Alleged commercial success

274. Patent Owner citing its interrogatories argued in the Facebook litigation that the '961 patent's solution for correcting a bias in the nonce generation method of NIST FIPS 186-2 (the DSA standard) was incorporated into the open-source OpenSSL software library, as well as the ANSI X9.62 and IEEE P1363a standards. Ex. 1027 at pp. 3-4. I did not see anything in the interrogatory responses (Exhibit 1027) that cited to evidence, particularly expert evidence, that supported this

argument. Moreover, as I explain above, Miyaji discloses claims 1-4, 15-18, and 23-26 the ’961 patent, and Miyaji in view of ordinary skill in the art, and/or Bellare in light of LEDA disclose claims 1-6, 15-20, and 23-28 of the ’961 patent; hence, any alleged commercial success for these claims in OpenSSL or the ANSI X9.62 and IEEE P1363a standards is more properly ascribed to the prior art of Miyaji and/or Bellare in light of LEDA rather than to the ’961 patent.

275. Further, OpenSSL is an open-source software package that provides many additional features and functionalities beyond the functionalities alleged by Patent Owner as being recited in the claims of the ’961 Patent. Exemplary additional features of OpenSSL include encryption/decryption algorithms such as AES, Blowfish, Camellia, Chacha20, Poly1305, SEED, CAST-128, DES, IDEA, RC2, RC4, RC5, Triple DES, GOST 28147-89, SM4, cryptographic hash functions such as MD5, MD4, MD2, SHA-1, SHA-2, SHA-3, RIPEMD-160, MDC-2, GOST R 34.11-94, BLAKE2, Whirlpool, SM3, public-key cryptography methods that include RSA, Diffie–Hellman key exchange, X25519, Ed25519, X448, Ed448, and SM2.[19] I see no analysis by the Patent Owner in the Facebook litigation that these other algorithms practiced any of the claims of the ’961 patent.

[19] *See, e.g.,* OpenSSL, https://en.wikipedia.org/wiki/OpenSSL, last visited August 23, 2020.

276. Thus, Patent Owner does not, in my opinion, establish a nexus between the benefits of the '961 patent and any alleged commercial success of OpenSSL, ANSI X9.62, and/or IEEE P1363a. Therefore, I conclude that any alleged commercial success of OpenSSL, ANSI X9.62, and/or IEEE P1363a does not provide sufficient evidence of secondary considerations of non-obviousness for the '961 patent to overcome my conclusions above regarding the obviousness of the identified claims of the '961 patent.

277. I reserve the right to opine on whether OpenSSL, ANSI X9.62, and/or IEEE P1363a practice any of the claims of the '961 patent should I be presented with evidence, such as relevant opinions from a technical expert.

ii. Alleged long felt but unresolved need

278. Patent Owner argued in the Facebook litigation that the bias in the method for generating the nonce k in the prior art FIPS 186-1 and 186-2 standards for DSA shows a long felt but unresolved need for the inventions of the '961 patent. Ex. 1027 at p. 4. However, as I establish above, even for the sake of argument assuming the claims of the '961 patent were to presuppose that there was a long-felt need to address bias from the calculation of a public key in the DSA standard of FIPS 186, that need was nonetheless met by prior art that also contemplated rejection sampling, such as Miyaji and/or Bellare in light of LEDA. Thus, there is no long felt

but unresolved need expressed in FIPS 186-1 and 186-2 for the inventions of '961 patent, because, e.g., this need was met in earlier prior art, reflected at least by Miyaji and/or Bellare in light of LEDA.

iii. Alleged failure of others

279. Patent owner in the Facebook litigation apparently cited the bias in the DSA key generation methods of the prior art FIPS 186-1 and FIPS 186-2 standards as alleged failure of others at reaching the inventions of the '961 Patent. Ex. 1027 at p. 4. However, as explained above, this alleged failure was cured by prior art evidenced by Miyaji and/or Bellare in light of LEDA long before the earliest priority date of the '961 patent.

280. Further, in my opinion Patent Owner in its arguments in the Facebook litigation failed to establish a nexus between the alleged failure of others and the benefits of the inventions of the '961 patent. For example, a POSA would recognize that the bias in the method of the FIPS 186-1 and 186-2 standards for generating the nonce k in the DSA signature standard can be easily remedied in ways that do not practice any of the claims of the '961 patent.

281. For instance, the FIPS 186-3 standard (published in June 2009) specifies two different ways to avoid the bias in the previous method for generating the nonce k for the DSA signature standard (given in the FIPS 186-1 and 186-2), and

Patent Owner apparently argued that neither of these methods practice any of the claims of the '961 patent.

282. The first way is to simply generate extra random bits and then do a reduction mod q-1:

Process:

1. N = **len**(q); L = **len**(p).

   Comment: Check that the (L, N) pair is specified in Section 4.2.

2. If the (L, N) pair is invalid, then return an **ERROR** indication, *Invalid_k*, and *Invalid_k_inverse*.
3. *requested_security_strength* = the security strength associated with the (L, N) pair; see SP 800-57.
4. Obtain a string of N+64 *returned_bits* from an **RBG** with a security strength of *requested_security_strength* or more. If an **ERROR** indication is returned, then return an **ERROR** indication, *Invalid_k*, and *Invalid_k_inverse*.
5. Convert *returned_bits* to the (non-negative) integer c (see Appendix C.2.1).
6. $k = (c \bmod (q-1)) + 1$.
7. ($status$, k^{-1}) = **inverse** (k, q).
8. Return *status*, k, and k^{-1}.

FIPS 186-3 (Ex. 1036) at p. 49 (highlighting added).

283. A POSA would understand that this method reduces the bias present in the nonce-generation method disclosed in FIPS 186-1 and 186-2 to be less than $1/2^{64}$, which a POSA would find to be negligible, e.g., it is less than one in a billion-billion, i.e., less than one in a quintillion (a 1 with 18 zeros after it). For reference, a quintillion seconds is roughly 32 billion years, which is more than twice the age of the universe.

284. Further, in IPR2019-00923, Patent Owner argued that the above method does not practice the steps of any of the claims of the '961 patent. Patent Owner's Preliminary Response in IPR2019-00923 (Ex. 1029) at pp. 36-37.

285. Another way is to use rejection sampling, but skip the use of a hash function (which was identified as "G( )" in FIPS 186-1 and 186-2):

Process:

1. N = **len**(q); L = **len**(p).

   Comment: Check that the (L, N) pair is specified in Section 4.2).
2. If the (L, N) pair is invalid, then return an **ERROR** indication, *Invalid_k*, and *Invalid_k_inverse*.
3. *requested_security_strength* = the security strength associated with the (L, N) pair; see SP 800-57.
4. Obtain a string of *N returned_bits* from an **RBG** with a security strength of *requested_security_strength* or more. If an **ERROR** indication is returned, then return an **ERROR** indication, *Invalid_k*, and *Invalid_k_inverse*.
5. Convert *returned_bits* to the (non-negative) integer c (see Appendix C.2.1).
6. If ($c > q–2$), then go to step 4.
7. $k = c + 1$.
8. $(status, k^{-1})$ = **inverse**(k, q).
9. Return *status*, k, and k^{-1}.

FIPS 186-3 (Ex. 1036) at p. 50 (highlighting added).

286. In IPR2019-0923, Patent Owner argued that this method does not practice the steps in any of the claims of the '961 patent, e.g., because it avoids use of the hash function, "G()." Indeed, Patent Owner argued that the use of such a hash function in the above method "would have been inefficient and unnecessary." Patent Owner's Preliminary Response in IPR2019-00923 (Ex. 1029) at pp. 37-38.

287. Thus, I conclude that the alleged failure of others, as evidenced in the bias for generating the nonce k in the DSA specification in FIPS 186-1 and 186-2 does not provide evidence for secondary considerations of non-obviousness for the '961 patent.

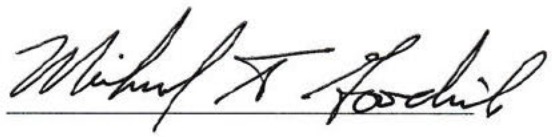
iv. Alleged copying of the invention by others

288. In the Facebook litigation, Patent Owner apparently cited OpenSSL as having copied the invention of the '961 Patent. IPR-Rogs (Ex. 1027) at p. 4. Patent owner provides no proof of such copying, or that this contention has been established in a court proceeding. Patent owner does not demonstrate how or why OpenSSL supposedly practices the invention of the '961 patent, let alone that the authors of OpenSLL copied the invention of the '961 patent. I therefore reserve the right to respond should such evidence of alleged copying be presented by the Patent Owner. For now, in my opinion, there is no evidence that establishes that the authors for OpenSSL copied the '961 patent.

Declaration

I declare that all statements made herein of my own knowledge are true, and that all statements made on information and belief are believed to be true, and that these statements were made with the knowledge that willful false statements and the like so made are punishable by fine or imprisonment, or both, under Section 1001 of Title 18 of the United States Code.

Executed at Irvine, California on October 2, 2020.



Michael T. Goodrich, Ph.D.

APPENDIX A

CURRICULUM VITAE

Michael T. Goodrich

Dept. of Computer Science
Bren School of Info. & Computer Sciences
University of California, Irvine
Irvine, CA 92697-3435

E-mail: goodrich (at) acm.org
http://www.ics.uci.edu/~goodrich/
Phone: (949)824-9366
Fax: (949)824-4056

CITIZENSHIP: U.S.A.

EDUCATION

Ph.D.	1987	*Efficient Parallel Techniques for Computational Geometry* Computer Science, Purdue Univ. (M.J. Atallah, advisor)
M.S.	1985	Computer Science, Purdue Univ.
B.A.	1983	Mathematics and Computer Science, Calvin Univ.

PROFESSIONAL EXPERIENCE

July '19 to present	Distinguished Professor, Dept. of Computer Science Univ. of California, Irvine
April '07 to June '19	Chancellor's Professor, Dept. of Computer Science Univ. of California, Irvine
July '12 to June '13	Chair, Dept. of Computer Science Univ. of California, Irvine
October '06 to June '12	Assoc. Dean for Faculty Dev., Bren School of Info. and Comp. Sci. Univ. of California, Irvine
July '01 to March '07	Professor, Dept. of Computer Science Univ. of California, Irvine
Fall '00	Visiting Professor of Computer Science Brown Univ.
July '96 to June '02	Professor of Computer Science (on leave, from July '01) Johns Hopkins Univ.
July '92 to June '96	Associate Professor of Computer Science Johns Hopkins Univ.
Spring '94	Visiting Associate Professor of Computer Science Univ. of Illinois, Urbana-Champaign
July '87 to June '92	Assistant Professor of Computer Science Johns Hopkins Univ.

RESEARCH INTERESTS

Algorithm and Data Structure Design
Information Assurance, Privacy, and Security
Principles of Database, Parallel, and Big-data Systems
Networks and Distributed Computing
Graph Visualization and Computational Geometry

PRIZES, HONORS, AND AWARDS

- *Compere Loveless Fellowship in Computer Sciences*, Purdue Univ., 1985
- *Research Initiation Award*, National Science Foundation, 1988
- *Oraculum Award for Excellence in Teaching*, Johns Hopkins, 1993, 1994, 1995
- *ACM Recognition of Service Award*, 1996
- *Robert B. Pond, Sr. Award for Excellence in Undergraduate Teaching*, Johns Hopkins, 1998
- *Elected Senior Member*, the Institute of Electrical and Electronics Engineers (IEEE), 1999

- *Spirit of Technology Transition Award*, DARPA Dynamic Coalitions Program, 2002
- *Brown Univ. Award for Technological Innovation* (with R. Tamassia, N. Triandopoulos, D. Yao, and D. Ellis), 2006
- *ACM Distinguished Scientist*, 2006
- *2006 IEEE Computer Society Technical Achievement Award*, "for outstanding contributions to the design of parallel and distributed algorithms for fundamental combinatorial and geometric problems"
- *Fulbright Scholar*, 2007, for senior specialist service to University of Aarhus, Denmark
- *Fellow of the San Diego Supercomputer Center*, 2007
- *Fellow of the American Association for the Advancement of Science (AAAS)*, "for distinguished contributions to parallel and distributed algorithms for combinatorial and geometric problems, and excellence in teaching, academic and professional service, and textbook writing," 2007
- *Named as Chancellor's Professor*, for "demonstrated unusual academic merit and whose continued promise for scholarly achievement is unusually high," Univ. of California, Irvine, 2007
- *Fellow of the Institute of Electrical and Electronics Engineers (IEEE)*, "for contributions to parallel and distributed algorithms for combinatorial and geometric problems," 2009
- *Fellow of the ACM*, "for contributions to data structures and algorithms for combinatorial and geometric problems," 2009
- *ICS Dean's Award for Research*, "for contributions in the area of parallel and distributed algorithms," 2014
- *Chancellor's Award for Excellence in Fostering Undergraduate Research*, Univ. of California, Irvine, 2016
- *Faculty Mentor of the Month*, Undergraduate Research Opportunities Program (UROP), Univ. of California, Irvine, April 2016
- *Elected as a foreign member*, Royal Danish Academy of Sciences and Letters, April 2018
- *Named as Distinguished Professor*, for achieving "the highest levels of scholarship" over the course of a career and having "earned national and international level distinctions and honors of the highest level," Univ. of California, Irvine, 2019

PUBLICATIONS

Google Scholar Citation Statistics:

- Total citations: over 14,500
- H-index (top H publications with at least H citations): 66
- i10-index (publications with at least 10 citations): 195

Patents and Patent Applications:

P-1. G. Ateniese, B. de Medeiros, and M.T. Goodrich, "Intermediated Delivery Scheme for Asymmetric Fair Exchange of Electronic Items," U.S. Patent Application US 2004/0073790 A1, April 15, 2004.

P-2. M.T. Goodrich and R. Tamassia, "Efficient Authenticated Dictionaries with Skip Lists and Commutative Hashing," U.S. Patent 7,257,711, August 14, 2007.

P-3. J.W. Green, J.L. Schultz, Y. Amir, and M.T. Goodrich, "High Refresh-Rate Retrieval of Freshly Published Content using Distributed Crawling," U.S. Patent 7,299,219, November 20, 2007.

P-4. R. Tamstorf, M.T. Goodrich, D. Eppstein, "Attribute Transfer Between Computer Models Including Identifying Isomorphic Regions in Polygonal Meshes," U.S. Patent 8,681,145, March 25, 2014. (also Application US 2010/0238166 A1, September 23, 2010).

P-5. N. Triandopoulos, M.T. Goodrich, D. Nguyen, O. Ohrimenko, C. Papamanthou, R. Tamassia, C.V. Lopes, "Techniques for Verifying Search Results Over a Distributed Collection," U.S. Patent, 9,152,716, October 6, 2015.

Books and Monographs:

B-1. M.T. Goodrich and R. Tamassia, *Data Structures and Algorithms in Java*, John Wiley and Sons, Inc., 1998.

B-2. M.T. Goodrich and C.C. McGeoch, eds., *Algorithm Engineering and Experimentation*, Lecture Notes in Computer Science (LNCS), Vol. 1619, Springer-Verlag, 1999.

B-3. M.T. Goodrich and R. Tamassia, *Data Structures and Algorithms in Java, Second Edition*, John Wiley and Sons, Inc., 2001.

B-4. M.T. Goodrich and R. Tamassia, *Algorithm Design: Foundations, Analysis, and Internet Examples*, John Wiley and Sons, Inc., 2002.

B-5. M.T. Goodrich and S.G. Kobourov, eds., *10th Int. Symp. on Graph Drawing* (GD), Lecture Notes in Computer Science, Vol. 2528, Springer-Verlag, 2002.

B-6. M.T. Goodrich, R. Tamassia, and D. Mount, *Data Structures and Algorithms in C++*, John Wiley and Sons, Inc., 2004.

B-7. M.T. Goodrich and R. Tamassia, *Data Structures and Algorithms in Java, Third Edition*, John Wiley and Sons, Inc., 2004.

B-8. M.T. Goodrich and R. Tamassia, *Data Structures and Algorithms in Java, Fourth Edition*, John Wiley and Sons, Inc., 2006.

B-9. M.T. Goodrich and R. Tamassia, *Data Structures and Algorithms in Java, Fifth Edition*, John Wiley and Sons, Inc., 2011.

B-10. M.T. Goodrich and R. Tamassia, *Introduction to Computer Security*, Addison-Wesley, Inc., 2011.

B-11. M.T. Goodrich, R. Tamassia, and D. Mount, *Data Structures and Algorithms in C++, Second Edition*, John Wiley and Sons, Inc., 2011.

B-12. M.T. Goodrich, R. Tamassia, and M. Goldwasser, *Data Structures and Algorithms in Python*, John Wiley and Sons, Inc., 2013.

B-13. M.T. Goodrich, R. Tamassia, and M. Goldwasser, *Data Structures and Algorithms in Java, Sixth Edition*, John Wiley and Sons, Inc., 2014.

B-14. M.T. Goodrich and R. Tamassia, *Algorithm Design and Applications*, Wiley, 2015.

Book Chapters:

Ch-1. M.J. Atallah and M.T. Goodrich, "Deterministic Parallel Computational Geometry," in *Synthesis of Parallel Algorithms*, J.H. Reif, ed., Morgan Kaufmann, 497–536, 1993.

Ch-2. M.T. Goodrich, "The Grand Challenges of Geometric Computing," in *Developing a Computer Science Agenda for High-Performance Computing*, U. Vishkin, ed., ACM Press, 64–68, 1994.

Ch-3. M.T. Goodrich, "Parallel Algorithms in Geometry," *CRC Handbook of Discrete and Computational Geometry*, J.E. Goodman and J. O'Rourke, eds., CRC Press, Inc., 669–682, 1997.

Ch-4. M.T. Goodrich and K. Ramaiyer, "Geometric Data Structures," *Handbook of Computational Geometry*, J.-R. Sack and J. Urrutia, eds., Elsevier Science Publishing, 463–489, 2000.

Ch-5. M.T. Goodrich and R. Tamassia, "Simplified Analyses of Randomized Algorithms for Searching, Sorting, and Selection," *Handbook of Randomized Computing*, S. Rajasekaran, P.M. Pardalos, J.H. Reif, and J.D.P. Rolim, eds., Kluwer Academic Publishers, Vol. 1, 23–34, 2001.

Ch-6. M.T. Goodrich, "Parallel Algorithms in Geometry," *Handbook of Discrete and Computational Geometry, Second Edition*, J.E. Goodman and J. O'Rourke, eds., Chapman & Hall/CRC Press, Inc., 953–967, 2004. (Revised version of Ch-3.)

Ch-7. C. Duncan and M.T. Goodrich, "Approximate Geometric Query Structures," *Handbook of Data Structures and Applications*, Chapman & Hall/CRC Press, Inc., 26-1–26-17, 2005.

Ch-8. M.T. Goodrich, R. Tamassia, and L. Vismara, "Data Structures in JDSL," *Handbook of Data Structures and Applications*, Chapman & Hall/CRC Press, Inc., 43-1–43-22, 2005.

Ch-9. Y. Cho, L. Bao and M.T. Goodrich, "Secure Location-Based Access Control in WLAN Systems," *From Problem Toward Solution: Wireless and Sensor Networks Security*, Zhen Jiang and Yi Pan, eds., Nova Science Publishers, Inc., Chapter 17, 2007.

Ch-10. M.T. Goodrich and M.J. Nelson, "Distributed Peer-to-Peer Data Structures," *Handbook of Parallel Computing: Models, Algorithms and Applications*, R. Rajasekaran and J. Reif, eds., CRC Press, 17-1–17-17, 2008.

Ch-11. C.A. Duncan and M.T. Goodrich, "Planar Orthogonal and Polyline Drawing Algorithms," *Handbook of Graph Drawing and Visualization*, CRC Press, Inc., 223–246, 2013.

Ch-12. M.T. Goodrich, R. Tamassia, and L. Vismara, "Data Structures in JDSL," *Handbook of Data Structures and Applications*, 2nd edition, Chapman and Hall/CRC, Taylor & Francis, Inc., 43-1–43-22, 2018.

Journal Papers:

J-1. M.J. Atallah and M.T. Goodrich, "Efficient Parallel Solutions to Some Geometric Problems," *Journal of Parallel and Distributed Computing*, **3**(4), 1986, 492–507.

J-2. M.T. Goodrich, "Finding the Convex Hull of a Sorted Point Set in Parallel," *Information Processing Letters*, **26**, 1987, 173–179.

J-3. H. ElGindy and M.T. Goodrich, "Parallel Algorithms for Shortest Path Problems in Polygons," *The Visual Computer*, **3**(6), 1988, 371–378.

J-4. M.J. Atallah and M.T. Goodrich, "Parallel Algorithms For Some Functions of Two Convex Polygons," *Algorithmica*, **3**, 1988, 535–548.

J-5. M.J. Atallah, R. Cole, and M.T. Goodrich, "Cascading Divide-and-Conquer: A Technique for Designing Parallel Algorithms," *SIAM Journal on Computing*, **18**(3), 1989, 499–532.

J-6. M.T. Goodrich, "Triangulating a Polygon in Parallel," *Journal of Algorithms*, **10**, 1989, 327–351.

J-7. M.T. Goodrich and M.J. Atallah, "On Performing Robust Order Statistics in Tree-Structured Dictionary Machines," *Journal of Parallel and Distributed Computing*, **9**(1), 1990, 69–76.

J-8. M.T. Goodrich and J.S. Snoeyink, "Stabbing Parallel Segments with a Convex Polygon," *Computer Vision, Graphics and Image Processing*, **49**, 1990, 152–170.

J-9. J. Johnstone and M.T. Goodrich, "A Localized Method for Intersecting Plane Algebraic Curve Segments," *The Visual Computer*, **7**(2–3), 1991, 60–71.

J-10. M.T. Goodrich, "Intersecting Line Segments in Parallel with an Output-Sensitive Number of Processors," *SIAM Journal on Computing*, **20**(4), 1991, 737–755.

J-11. R. Cole and M.T. Goodrich, "Optimal Parallel Algorithms for Point-Set and Polygon Problems," *Algorithmica*, **7**, 1992, 3–23.

J-12. M.T. Goodrich, "A Polygonal Approach to Hidden-Line and Hidden-Surface Elimination," *Computer Vision, Graphics, and Image Processing: Graphical Models and Image Processing*, **54**(1), 1992, 1–12.

J-13. M.T. Goodrich, S. Shauck, and S. Guha, "Parallel Methods for Visibility and Shortest Path Problems in Simple Polygons," *Algorithmica*, **8**, 1992, 461–486, with addendum in *Algorithmica*, **9**, 1993, 515–516.

J-14. M.T. Goodrich, C. Ó'Dúnlaing, and C. Yap "Constructing the Voronoi Diagram of a Set of Line Segments in Parallel," *Algorithmica*, **9**, 1993, 128–141.

J-15. M.T. Goodrich, "Constructing the Convex Hull of a Partially Sorted Set of Points," *Computational Geometry: Theory and Applications*, **2**, 1993, 267–278.

J-16. M.T. Goodrich, "Constructing Arrangements Optimally in Parallel," *Discrete and Computational Geometry*, **9**, 1993, 371–385.

J-17. M.T. Goodrich, M.J. Atallah, and M. Overmars, "Output-Sensitive Methods for Rectilinear Hidden Surface Removal," *Information and Computation*, **107**(1), 1993, 1–24.

J-18. M.J. Atallah, P. Callahan, and M.T. Goodrich, "P-Complete Geometric Problems," *Int. Journal of Computational Geometry & Applications*, **3**(4), 1993, 443–462.

J-19. M.J. Atallah, M.T. Goodrich, and S.R. Kosaraju, "Parallel Algorithms for Evaluating Sequences of Set-Manipulation Operations," *Journal of the ACM*, **41**(6), 1994, 1049–1088.

J-20. M.T. Goodrich, "Efficient Piecewise-Linear Function Approximation Using the Uniform Metric," *Discrete and Computational Geometry*, **14**, 1995, 445–462.

J-21. H. Brönnimann and M.T. Goodrich, "Almost Optimal Set Covers in Finite VC-Dimension," *Discrete and Computational Geometry*, **14**, 1995, 463–479.

J-22. M.T. Goodrich, "Planar Separators and Parallel Polygon Triangulation," *J. Computer and System Sciences*, **51**(3), 1995, 374–389.

J-23. M.T. Goodrich, M. Ghouse, and J. Bright, "Sweep Methods for Parallel Computational Geometry," *Algorithmica*, **15**(2), 1996, 126–153.

J-24. M.T. Goodrich and S.R. Kosaraju, "Sorting on a Parallel Pointer Machine with Applications to Set Expression Evaluation," *Journal of the ACM*, **43**(2), 1996, 331–361.

J-25. A. Garg, M.T. Goodrich, and R. Tamassia, "Planar Upward Tree Drawings with Optimal Area," *International Journal of Computational Geometry & Applications*, **6**(3), 1996, 333–356.

J-26. M.H. Nodine, M.T. Goodrich, and J.S. Vitter, "Blocking for External Graph Searching," *Algorithmica*, **16**(2), 1996, 181–214.

J-27. R. Cole, M.T. Goodrich, C. Ó Dúnlaing, "A Nearly Optimal Deterministic Parallel Voronoi Diagram Algorithm," *Algorithmica*, **16**, 1996, 569–617.

J-28. G. Das and M.T. Goodrich, "On the Complexity of Optimization Problems for 3-Dimensional Convex Polyhedra and Decision Trees," *Computational Geometry: Theory and Applications*, **8**, 1997, 123–137.

J-29. M.T. Goodrich and R. Tamassia, "Dynamic Ray Shooting and Shortest Paths in Planar Subdivisions via Balanced Geodesic Triangulations," *J. Algorithms*, **23**, 1997, 51–73.

J-30. M. Ghouse and M.T. Goodrich, "Fast Randomized Parallel Methods for Planar Convex Hull Construction," *Computational Geometry: Theory and Applications*, **7**, 1997, 219–235.

J-31. L.P. Chew, M.T. Goodrich, D.P. Huttenlocher, K. Kedem, J.M. Kleinberg, and D. Kravets, "Geometric Pattern Matching under Euclidean Motion," *Computational Geometry: Theory and Applications*, **7**, 1997, 113-124.

J-32. M.T. Goodrich and E.A. Ramos, "Bounded-Independence Derandomization of Geometric

Partitioning with Applications to Parallel Fixed-Dimensional Linear Programming," *Discrete & Computational Geometry*, **18**(4), 1997, 397–420.

J-33. M.T. Goodrich, "An Improved Ray Shooting Method for Constructive Solid Geometry Models via Tree Contraction," *International Journal of Computational Geometry & Applications*, **8**(1), 1998, 1–23.

J-34. G. Barequet, A.J. Briggs, M.T. Dickerson, and M.T. Goodrich, "Offset-Polygon Annulus Placement Problems," *Computational Geometry: Theory and Applications*, **11**(3–4), 1998–99, 125–141.

J-35. M.T. Goodrich and R. Tamassia, "Dynamic Trees and Dynamic Point Location," *SIAM J. Comput.*, **28**(2), 1999, 612–636.

J-36. G. Barequet, S.S. Bridgeman, C.A. Duncan, M.T. Goodrich, and R. Tamassia, "GeomNet: Geometric Computing Over the Internet," *IEEE Internet Computing*, **3**(2), 1999, 21–29.

J-37. M.T. Goodrich, J.S.B. Mitchell, and M.W. Orletsky, "Approximate Geometric Pattern Matching Under Rigid Motion," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, **21**(4), 1999, 371–379.

J-38. M.T. Goodrich, "Communication-Efficient Parallel Sorting," *SIAM Journal on Computing*, **29**(2), 1999, 416–432.

J-39. C.A. Duncan, M.T. Goodrich, S.G. Kobourov, "Balanced Aspect Ratio Trees and Their Use for Drawing Very Large Graphs," *Journal of Graph Algorithms and Applications*, **4**(3), 2000, 19–46. Also available at www.cs.brown.edu/publications/jgaa/.

J-40. M.T. Goodrich and C.G. Wagner, "A Framework for Drawing Planar Graphs with Curves and Polylines," *Journal of Algorithms*, **37**, 2000, 399–421.

J-41. C.A. Duncan, M.T. Goodrich, S.G. Kobourov, "Balanced Aspect Ratio Trees: Combining the Benefits of k-D Trees and Octrees," *J. Algorithms*, **38**, 2001, 303–333.

J-42. G. Barequet, M. Dickerson, and M.T. Goodrich, "Voronoi Diagrams for Polygon-Offset Distance Functions," *Discrete and Computational Geometry*, **25**(2), 2001, 271–291.

J-43. C.C. Cheng, C.A. Duncan, M.T. Goodrich, and S.G. Kobourov, "Drawing Planar Graphs with Circular Arcs," *Discrete and Computational Geometry*, **25**(3), 2001, 405–418.

J-44. N.M. Amato, M.T. Goodrich, and E.A. Ramos, "A Randomized Algorithm for Triangulating a Simple Polygon in Linear Time," *Discrete and Computational Geometry*, **26**(2), 2001, 245–265.

J-45. R. Tamassia, M.T. Goodrich, L. Vismara, M. Handy, G. Shubina, R. Cohen, B. Hudson, R.S. Baker, N. Gelfand, and U. Brandes, "JDSL: The Data Structures Library in Java," *Dr. Dobbs Journal*, **323**, 2001, 21–31.

J-46. G. Barequet, D.Z. Chen, O. Daescu, M.T. Goodrich, and J.S. Snoeyink, "Efficiently Approximating Polygonal Paths in Three and Higher Dimensions," *Algorithmica*, **33**(2), 2002, 150–167.

J-47. T. Chan, M.T. Goodrich, S.R. Kosaraju, and R. Tamassia, "Optimizing Area and Aspect Ratio in Straight-Line Orthogonal Tree Drawings," *Computational Geometry: Theory and Applications*, **23**(2), 2002, 153–162.

J-48. C.A. Duncan, M.T. Goodrich, and S.G. Kobourov, "Planarity-Preserving Clustering and Embedding for Large Planar Graphs," *Computational Geometry: Theory and Applications*, **24**(2), 2003, 95–114.

J-49. A.L. Buchsbaum and M.T. Goodrich, "Three-Dimensional Layers of Maxima," *Algorithmica*, **39**, 2004, 275–286.

J-50. G. Barequet, M.T. Goodrich, and C. Riley, "Drawing Graphs with Large Vertices and Thick

Edges," *J. of Graph Algorithms and Applications* (JGAA), **8**(1), 2004, 3–20.

J-51. G. Barequet, M.T. Goodrich, A. Levi-Steiner, and D. Steiner, "Contour Interpolation by Straight Skeletons," *Graphical Models* (GM), **66**(4), 2004, 245–260.

J-52. P. Gajer, M.T. Goodrich, and S.G. Kobourov, "A Multi-Dimensional Approach to Force-Directed Layouts of Large Graphs," *Computational Geometry: Theory and Applications*, **29**(1), 3–18, 2004.

J-53. G. Barequet, P. Bose, M.T. Dickerson, and M.T. Goodrich, "Optimizing a Constrained Convex Polygonal Annulus," *J. of Discrete Algorithms* (JDA), **3**(1), 1–26, 2005.

J-54. A. Bagchi, A.L. Buchsbaum, and M.T. Goodrich, "Biased Skip Lists," *Algorithmica*, **42**(1), 31–48, 2005.

J-55. M. Dickerson, D. Eppstein, M.T. Goodrich, J. Meng, "Confluent Drawings: Visualizing Non-planar Diagrams in a Planar Way," *J. of Graph Algorithms and Applications* (JGAA), **9**(1), 31–52, 2005.

J-56. A. Bagchi, A. Chaudhary, M.T. Goodrich, C. Li, and M. Shmueli-Scheuer, "Achieving Communication Efficiency through Push-Pull Partitioning of Semantic Spaces to Disseminate Dynamic Information," *IEEE Trans. on Knowledge and Data Engineering* (TKDE), **18**(10), 1352–1367, 2006.

J-57. D. Eppstein, M.T. Goodrich, and J.Y. Meng, "Confluent Layered Drawings," *Algorithmica*, **47**(4), 439–452, 2007.

J-58. A. Bagchi, A. Chaudhary, D. Eppstein, and M.T. Goodrich, "Deterministic Sampling and Range Counting in Geometric Data Streams," *ACM Transactions on Algorithms*, **3**(2), Article 16, 2007, 18 pages.

J-59. D. Eppstein, M.T. Goodrich, and D. Hirschberg, "Improved Combinatorial Group Testing Algorithms for Real-World Problem Sizes," *SIAM Journal on Computing*, **36**(5), 1360–1375, 2007.

J-60. D. Eppstein, M.T. Goodrich, and J.Z. Sun, "Skip Quadtrees: Dynamic Data Structures for Multidimensional Point Sets," *Int. Journal on Computational Geometry and Applications*, **18**(1/2), 131–160, 2008.

J-61. M.T. Goodrich, "Probabilistic Packet Marking for Large-Scale IP Traceback," *IEEE/ACM Transactions on Networking*, **16**(1), 15–24, 2008.

J-62. M.T. Goodrich and D.S. Hirschberg, "Improved Adaptive Group Testing Algorithms with Applications to Multiple Access Channels and Dead Sensor Diagnosis," *Journal of Combinatorial Optimization*, **15**(1), 95–121, 2008.

J-63. M.T. Goodrich, R. Tamassia, and D. Yao, "Notarized Federated ID Management and Authentication," *Journal of Computer Security*, **16**(4), 399–418, 2008.

J-64. M.T. Goodrich, "Pipelined Algorithms to Detect Cheating in Long-Term Grid Computations," *Theoretical Computer Science*, **408**, 199–207, 2008.

J-65. D. Eppstein, M.T. Goodrich, E. Kim, and R. Tamstorf, "Motorcycle Graphs: Canonical Quad Mesh Partitioning," *Computer Graphics Forum*, special issue on papers from 6th European Symp. on Geometry Processing (SGP), **27**(6), 1477–1486, 2008.

J-66. M.T. Goodrich, M. Sirivianos, J. Solis, C. Soriente, G. Tsudik, E. Uzun, "Using Audio in Secure Device Pairing," *Int. J. Security and Networks*, **4**(1/2), 57–68, 2009.

J-67. M.T. Goodrich, "On the Algorithmic Complexity of the Mastermind Game with Black-Peg Results," *Information Processing Letters*, **109**, 675–678, 2009.

J-68. D. Eppstein, M.T. Goodrich, E. Kim, and R. Tamstorf, "Approximate Topological Matching of Quad Meshes," *The Visual Computer*, **25**(8), 771–783, 2009.

J-69. D. Eppstein and M.T. Goodrich, "Succinct Greedy Geometric Routing Using Hyperbolic Geometry," *IEEE Transactions on Computers*, **60**(11), 1571–1580, 2011. Posted online Dec. 2010, IEEE Computer Society Digital Library.

J-70. D. Eppstein, M.T. Goodrich, and D. Strash, "Linear-Time Algorithms for Geometric Graphs with Sublinearly Many Edge Crossings," *SIAM Journal on Computing*, **39**(8), 3814–3829. 2010.

J-71. M.T. Goodrich, R. Tamassia, and N. Triandopoulos, "Efficient Authenticated Data Structures for Graph Connectivity and Geometric Search Problems," *Algorithmica*, **60**(3), 505–552, 2011.

J-72. D. Eppstein and M.T. Goodrich, "Straggler Identification in Round-Trip Data Streams via Newton's Identities and Invertible Bloom Filters," *IEEE Transactions on Knowledge and Data Engineering* (TKDE), **23**(2), 297–306, 2011.

J-73. C.A. Duncan, M.T. Goodrich, S.G. Kobourov, "Planar Drawings of Higher-Genus Graphs," *Journal of Graph Algorithms and Applications*, **15**(1), 7–32, 2011.

J-74. M.T. Dickerson, M.T. Goodrich, T.D. Dickerson, and Y.D. Zhuo "Round-Trip Voronoi Diagrams and Doubling Density in Geographic Networks," *Transactions on Computational Science*, M.L. Gavrilova et al. (Eds.), Vol. 14, LNCS 6970, 211–238, 2011.

J-75. M.T. Goodrich, "Randomized Shellsort: A Simple Data-Oblivious Sorting Algorithm," *Journal of the ACM*, **58**(6), Article No. 27, 2011.

J-76. C.A. Duncan, D. Eppstein, M.T. Goodrich, S. Kobourov, and M. Nöllenburg, "Lombardi Drawings of Graphs," *Journal of Graph Algorithms and Applications (JGAA)*, **16**(1), 85–108, 2012.

J-77. E. Wolf-Chambers, D. Eppstein, M.T. Goodrich, and M. Löffler, "Drawing Graphs in the Plane with a Prescribed Outer Face and Polynomial Area," *Journal of Graph Algorithms and Applications (JGAA)*, **16**(2), 243–259, 2012.

J-78. M.T. Goodrich, D. Nguyen, O. Ohrimenko, C. Papamanthou, R. Tamassia, N. Triandopoulos, and C.V. Lopes, "Efficient Verification of Web-Content Searching Through Authenticated Web Crawlers," *Proc. VLDB*, **5**(10):920-931, 2012.

J-79. D. Eppstein, M.T. Goodrich, D. Strash, and L. Trott, "Extended Dynamic Subgraph Statistics Using h-Index Parameterized Data Structures," *Theoretical Computer Science*, **447**, 44–52, 2012.

J-80. M.T. Goodrich, "Learning Character Strings via Mastermind Queries, With a Case Study Involving mtDNA," *IEEE Transactions on Information Theory*, **58**(11), 6726–6736, 2012.

J-81. A.U. Asuncion and M.T. Goodrich, "Nonadaptive Mastermind Algorithms for String and Vector Databases, with Case Studies," *IEEE Transactions on Knowledge and Data Engineering* (TKDE), **25**(1), 131–144, 2013.

J-82. C.A. Duncan, D. Eppstein, M.T. Goodrich, S. Kobourov, and M. Nöllenburg, "Drawing Trees with Perfect Angular Resolution and Polynomial Area," *Discrete & Computational Geometry*, **49**(2), 157–182, 2013.

J-83. E. Angelino, M.T. Goodrich, M. Mitzenmacher and J. Thaler, "External Memory Multimaps," *Algorithmica*, **67**(1), 23–48, 2013.

J-84. D. Eppstein, M.T. Goodrich, M. Löffler, D. Strash and L. Trott, "Category-Based Routing in Social Networks: Membership Dimension and the Small-World Phenomenon," *Theoretical Computer Science*, **514**, 96–104, 2013.

J-85. M.T. Goodrich, "Spin-the-bottle Sort and Annealing Sort: Oblivious Sorting via Round-robin Random Comparisons," *Algorithmica*, **68**(4), 835–858, 2014.

J-86. Michael J. Bannister, William E. Devanny, David Eppstein, and M.T. Goodrich, "The Galois Complexity of Graph Drawing: Why Numerical Solutions are Ubiquitous for Force-Directed, Spectral, and Circle Packing Drawings," *Journal of Graph Algorithms and Applications*, **19**(2), 619–656, 2015.

J-87. C. Duncan, D. Eppstein, M.T. Goodrich, S.G. Kobourov and M. Löffler, "Planar and Poly-Arc Lombardi Drawings," *Journal of Computational Geometry* (JoCG), **9**(1), 328–355, 2018.

J-88. Gill Barequet, David Eppstein, M.T. Goodrich, and Nil Mamano, "Stable-Matching Voronoi Diagrams: Combinatorial Complexity and Algorithms," *Journal of Computational Geometry* (JoCG), **11**(1), 26–59, 2020.

Papers in Peer-Reviewed Proceedings:

C-1. M.J. Atallah and M.T. Goodrich, "Efficient Parallel Solutions to Geometric Problems," *1985 IEEE Int. Conf. on Parallel Processing (ICPP)*, 411–417. (Preliminary version of J-1.)

C-2. F. Berman, M.T. Goodrich, C. Koelbel, W. Robison, and K. Showell, "Prep-P: A Mapping Preprocessor for CHiP Computers," *1985 IEEE Int. Conf. on Parallel Processing*, 731–733.

C-3. M.J. Atallah and M.T. Goodrich, "Parallel Algorithms For Some Functions of Two Convex Polygons," *24th Allerton Conf. on Communication, Control and Computing*, 1986, 758–767. (Preliminary version of J-4.)

C-4. M.J. Atallah and M.T. Goodrich, "Efficient Plane Sweeping in Parallel," *2nd ACM Symp. on Computational Geometry* (SoCG), 1986, 216–225.

C-5. M.T. Goodrich, "A Polygonal Approach to Hidden-Line Elimination," *25th Allerton Conf. on Communication, Control, and Computing*, 1987, 849–858. (Preliminary version of J-12.)

C-6. M.J. Atallah, R. Cole, and M.T. Goodrich, "Cascading Divide-and-Conquer: A Technique for Designing Parallel Algorithms," *28th IEEE Symp. on Foundations of Computer Science* (FOCS), 1987, 151-160. (Preliminary version of J-5.)

C-7. M.J. Atallah, M.T. Goodrich, and S.R. Kosaraju, "Parallel Algorithms for Evaluating Sequences of Set-Manipulation Operations," *3rd Aegean Workshop on Computing* (AWOC), *Lecture Notes in Computer Science* (LNCS): 319, Springer-Verlag, 1988, 1–10. (Preliminary version of J-19.)

C-8. R. Cole and M.T. Goodrich, "Optimal Parallel Algorithms for Polygon and Point-Set Problems," *4th ACM Symp. on Computational Geometry* (SoCG), 1988, 201–210. (Preliminary version of J-11.)

C-9. M.T. Goodrich, "Intersecting Line Segments in Parallel with an Output-Sensitive Number of Processors," *1989 ACM Symp. on Parallel Algorithms and Architectures* (SPAA), 127–137. (Preliminary version of J-10.)

C-10. M.T. Goodrich and S.R. Kosaraju, "Sorting on a Parallel Pointer Machine with Applications to Set Expression Evaluation," *30th IEEE Symp. on Foundations of Computer Science* (FOCS), 1989, 190–195. (Preliminary version of J-24.)

C-11. M.T. Goodrich, C. Ó'Dúnlaing, and C. Yap "Constructing the Voronoi Diagram of a Set of Line Segments in Parallel," *Lecture Notes in Computer Science 382, Algorithms and Data Structures* (WADS), Springer-Verlag, 1989, 12–23. (Preliminary version of J-14.)

C-12. M.T. Goodrich and J.S. Snoeyink, "Stabbing Parallel Segments with a Convex Polygon," *Lecture Notes in Computer Science 382, Algorithms and Data Structures* (WADS), Springer-Verlag, 1989, 231–242. (Preliminary version of J-8.)

C-13. J. Johnstone and M.T. Goodrich, "A Localized Method for Intersecting Plane Algebraic Curve Segments," *New Advances in Computer Graphics: Proc. of Computer Graphics International '89*, R.A. Earnshaw, B. Wyvel, eds., Springer-Verlag, 1989, 165–181.

(Preliminary version of J-9.)

C-14. M.J. Atallah, P. Callahan, and M.T. Goodrich, "P-Complete Geometric Problems," *2nd ACM Symp. on Parallel Algorithms and Architectures* (SPAA), 1990, 317–326. (Preliminary version of J-18.)

C-15. R. Cole, M.T. Goodrich, C. Ó Dúnlaing, "Merging Free Trees in Parallel for Efficient Voronoi Diagram Construction", *17th Int. Conf. on Automata, Languages, and Programming* (ICALP), 1990, 432–445. (Preliminary version of J-27.)

C-16. M.T. Goodrich, M.J. Atallah, and M. Overmars, "An Input-Size/Output-Size Trade-Off in the Time-Complexity of Rectilinear Hidden-Surface Removal", *17th Int. Conf. on Automata, Languages, and Programming* (ICALP), 1990, 689–702. (Preliminary version of J-17.)

C-17. M.T. Goodrich, M. Ghouse, and J. Bright, "Generalized Sweep Methods for Parallel Computational Geometry," *2nd ACM Symp. on Parallel Algorithms and Architectures* (SPAA), 1990, 280–289. (Preliminary version of J-23.)

C-18. M.T. Goodrich, "Applying Parallel Processing Techniques to Classification Problems in Constructive Solid Geometry," *1st ACM-SIAM Symp. on Discrete Algorithms* (SODA), 1990, 118–128. (Preliminary version of J-33.)

C-19. M.T. Goodrich, S. Shauck, and S. Guha, "Parallel Methods for Visibility and Shortest Path Problems in Simple Polygons," *6th ACM Symp. on Computational Geometry* (SoCG), 1990, 73–82. (Preliminary version of J-13.)

C-20. M. Ghouse and M.T. Goodrich, "In-Place Techniques for Parallel Convex Hull Algorithms," *3rd ACM Symp. on Parallel Algorithms and Architectures* (SPAA), 1991, 192–203. (Preliminary version of J-30.)

C-21. M.T. Goodrich, "Constructing Arrangements Optimally in Parallel," *3rd ACM Symp. on Parallel Algorithms and Architectures* (SPAA), 1991, 169–179. (Preliminary version of J-16.)

C-22. M.T. Goodrich and R. Tamassia, "Dynamic Trees and Dynamic Point Location," *23rd ACM Symp. on Theory of Computing* (STOC), 1991, 523–533. (Preliminary version of J-35.)

C-23. M.T. Goodrich, "Using Approximation Algorithms to Design Parallel Algorithms that May Ignore Processor Allocation," *32nd IEEE Symp. on Foundations of Computer Science* (FOCS), 1991, 711–722.

C-24. M.T. Goodrich, "Planar Separators and Parallel Polygon Triangulation," *24th ACM Symp. on Theory of Computing* (STOC), 1992, 507–516. (Preliminary version of J-22.)

C-25. M.T. Goodrich, Y. Matias, U. Vishkin, "Approximate Parallel Prefix Computation and Its Applications," *7th IEEE Int. Parallel Processing Symp* (IPPS), 1993, 318–325.

C-26. M. Ghouse and M.T. Goodrich, "Experimental Evidence for the Power of Random Sampling in Practical Parallel Algorithms," *7th IEEE Int. Parallel Processing Symp* (IPPS), 1993, 549–556.

C-27. L.P. Chew, M.T. Goodrich, D.P. Huttenlocher, K. Kedem, J.M. Kleinberg, and D. Kravets, "Geometric Pattern Matching under Euclidean Motion," *5th Canadian Conf. on Computational Geometry* (CCCG), 1993, 151–156. (Preliminary version of J-31.)

C-28. M.T. Goodrich, "Geometric Partitioning Made Easier, Even in Parallel," *9th ACM Symp. on Computational Geometry* (SoCG), 1993, 73–82.

C-29. M.T. Goodrich and R. Tamassia, "Dynamic Ray Shooting and Shortest Paths via Balanced Geodesic Triangulations," *9th ACM Symp. on Computational Geometry* (SoCG), 1993, 318–327. (Preliminary version of J-29.)

C-30. A. Garg, M.T. Goodrich, and R. Tamassia, "Area-Efficient Upward Tree Drawings," *9th ACM Symp. on Computational Geometry* (SoCG), 1993, 359–368. (Preliminary version of J-25.)

C-31. M.H. Nodine, M.T. Goodrich, and J.S. Vitter, "Blocking for External Graph Searching," *12th ACM Symp. on Principles of Database Systems* (PODS), 1993, 222–232. (Preliminary version of J-26.)

C-32. E.M. Arkin, M.T. Goodrich, J.S.B. Mitchell, D. Mount, and S.S. Skiena, "Point Probe Decision Trees for Geometric Concept Classes," *Lecture Notes in Computer Science 709: Algorithms and Data Structures* (WADS), Springer-Verlag, 1993, 95–106.

C-33. M.T. Goodrich, J.J. Tsay, D.E. Vengroff, and J.S. Vitter, "External-Memory Computational Geometry," *34th IEEE Symp. on Foundations of Computer Science* (FOCS), 1993, 714–723.

C-34. M.T. Goodrich, Y. Matias, and U. Vishkin, "Optimal Parallel Approximation Algorithms for Prefix Sums and Integer Sorting," *5th ACM-SIAM Symp. on Discrete Algorithms* (SODA), 1994, 241–250.

C-35. H. Brönnimann and M.T. Goodrich, "Almost Optimal Set Covers in Finite VC-Dimension," *10th ACM Symp. on Computational Geometry* (SoCG), 1994, 293–302. (Preliminary version of J-21.)

C-36. M.T. Goodrich, "Efficient Piecewise-Linear Function Approximation Using the Uniform Metric," *10th ACM Symp. on Computational Geometry* (SoCG), 1994, 322–331. (Preliminary version of J-20.)

C-37. M.J. Atallah, M.T. Goodrich, and K. Ramaiyer, "Biased Finger Trees and Three-Dimensional Layers of Maxima," *10th ACM Symp. on Computational Geometry* (SoCG), 1994, 150–159.

C-38. M.T. Goodrich, J.S.B. Mitchell, and M.W. Orletsky, "Practical Methods for Approximate Geometric Pattern Matching under Rigid Motions," *10th ACM Symp. on Computational Geometry* (SoCG), 1994, 103–112. (Preliminary version of J-37.)

C-39. N.M. Amato, M.T. Goodrich, E.A. Ramos, "Parallel Algorithms for Higher-Dimensional Convex Hulls," *35th IEEE Symp. on Foundations of Computer Science* (FOCS), 1994, 683–694.

C-40. P.J. Tanenbaum, M.T. Goodrich, and E.R. Scheinerman, "Characterization and Recognition of Point-Halfspace and Related Orders," *2nd Int. Symp. on Graph Drawing* (GD), Lecture Notes in Computer Science 894, Springer-Verlag, 1994, 234–245.

C-41. Y.J. Chiang, M.T. Goodrich, E.F. Grove, R. Tamassia, D.E. Vengroff, and J.S. Vitter, "External-Memory Graph Algorithms," *6th ACM-SIAM Symp. on Discrete Algorithms* (SODA), 1995, 139–149.

C-42. N.M. Amato, M.T. Goodrich, and E.A. Ramos, "Computing Faces in Segment and Simplex Arrangements," *27th ACM Symp. on Theory of Computing* (STOC), 1995, 672–682.

C-43. P. Callahan, M.T. Goodrich, and K. Ramaiyer, "Topology B-Trees and Their Applications," *1995 Workshop on Algorithms and Data Structures* (WADS), Lecture Notes in Computer Science 955, Springer-Verlag, 381–392.

C-44. G. Das and M.T. Goodrich, "On the Complexity of Approximating and Illuminating Three-Dimensional Convex Polyhedra," *1995 Workshop on Algorithms and Data Structures* (WADS), Lecture Notes in Computer Science 955, Springer-Verlag, 74–85. (Preliminary version of J-28.)

C-45. M.T. Goodrich, "Fixed-Dimensional Parallel Linear Programming via Relative ϵ-Approximations," *7th ACM-SIAM Symp. on Discrete Algorithms* (SODA), 1996, 132–141. (Preliminary version of J-32.)

C-46. M. Chrobak, M.T. Goodrich, and R. Tamassia, "Convex Drawings of Graphs in Two and Three Dimensions," *12th ACM Symp. on Computational Geometry* (SoCG), 1996, 319–328.

C-47. M.T. Goodrich, "Communication-Efficient Parallel Sorting," *28th ACM Symp. on Theory of*

Computing (STOC), 1996, 247–256. (Preliminary version of J-38.)

C-48. T. Chan, M.T. Goodrich, S.R. Kosaraju, and R. Tamassia, "Optimizing Area and Aspect Ratio in Straight-Line Orthogonal Tree Drawings," *4th Int. Symp. on Graph Drawing* (GD), Lecture Notes in Computer Science 1190, Springer-Verlag, 1996, 63–75. (Preliminary version of J-47.)

C-49. M.T. Goodrich, "Randomized Fully-Scalable BSP Techniques for Multi-Searching and Convex Hull Construction," *8th ACM-SIAM Symp. on Discrete Algorithms* (SODA), 1997, 767–776.

C-50. C.A. Duncan, M.T. Goodrich, and E.A. Ramos, "Efficient Approximation and Optimization Algorithms for Computational Metrology," *8th ACM-SIAM Symp. on Discrete Algorithms* (SODA), 1997, 121–130.

C-51. M.T. Goodrich, M. Orletsky, and K. Ramaiyer, "Methods for Achieving Fast Query Times in Point Location Data Structures," *8th ACM-SIAM Symp. on Discrete Algorithms* (SODA), 1997, 757–766.

C-52. M.T. Goodrich, L.J. Guibas, J. Hershberger, P.J. Tanenbaum, "Snap Rounding Line Segments Efficiently in Two and Three Dimensions," *13th ACM Symp. on Computational Geometry* (SoCG), 1997, 284–293.

C-53. G. Barequet, S.S. Bridgeman, C.A. Duncan, M.T. Goodrich, and R. Tamassia, "Classical Computational Geometry in GeomNet," *13th ACM Symp. on Computational Geometry* (SoCG), 1997, 412–414.

C-54. G. Barequet, A. Briggs, M. Dickerson, C. Dima, and M.T. Goodrich, "Animating the Polygon-Offset Distance Function," *13th ACM Symp. on Computational Geometry* (SoCG), 1997, 479–480, and the *Video Review for the 13th ACM Symp. on Computational Geometry* (SoCG).

C-55. G. Barequet, A. Briggs, M. Dickerson, and M.T. Goodrich, "Offset-Polygon Annulus Placement Problems," *1997 Workshop on Algorithms and Data Structures* (WADS), 1997, 378–391. (Preliminary version of J-34.)

C-56. G. Barequet, M. Dickerson, and M.T. Goodrich, "Voronoi Diagrams for Polygon-Offset Distance Functions," *1997 Workshop on Algorithms and Data Structures* (WADS), 1997, 200–209. (Preliminary version of J-42.)

C-57. N. Gelfand, M.T. Goodrich, and R. Tamassia, "Teaching Data Structure Design Patterns," *29th ACM SIGCSE Technical Symp. on Computer Science Education*, 1998, 331–335.

C-58. M.T. Goodrich and R. Tamassia, "Teaching the Analysis of Algorithms with Visual Proofs," *29th ACM SIGCSE Technical Symp. on Computer Science Education*, 1998, 207–211.

C-59. G. Barequet, D.Z. Chen, O. Daescu, M.T. Goodrich, and J.S. Snoeyink, "Efficiently Approximating Polygonal Paths in Three and Higher Dimensions," *1998 ACM Symp. on Computational Geometry* (SoCG), 1998, 317–326. (Preliminary version of J-46.)

C-60. M.T. Goodrich and C.G. Wagner, "A Framework for Drawing Planar Graphs with Curves and Polylines," *6th Int. Symp. on Graph Drawing* (GD), Lecture Notes in Computer Science 1547, Springer-Verlag, 1998, 153–166. (Preliminary version of J-40.)

C-61. C.A. Duncan, M.T. Goodrich, S.G. Kobourov, "Balanced Aspect Ratio Trees and Their Use for Drawing Very Large Graphs," *6th Int. Symp. on Graph Drawing* (GD), Lecture Notes in Computer Science 1547, Springer-Verlag, 1998, 111–124. (Preliminary version of J-39.)

C-62. M.T. Goodrich, M. Handy, B. Hudson, and R. Tamassia, "Abstracting Positional Information in Data Structures: Locators and Positions in JDSL," *Object-Oriented Programming, Systems, Languages & Applications (OOPSLA) '98 Technical Notes*, 1998.

C-63. M.T. Goodrich and J.G. Kloss II, "Tiered Vector: An Efficient Dynamic Array for JDSL," *Object-Oriented Programming, Systems, Languages & Applications (OOPSLA) '98 Technical Notes*, 1998.

C-64. M.T. Goodrich, M. Handy, B. Hudson, and R. Tamassia, "Accessing the Internal Organization of Data Structures in the JDSL Library," *Int. Workshop on Algorithm Engineering and Experimentation* (ALENEX), Springer-Verlag, Lecture Notes in Computer Science, Vol. 1619, 1999, 124–139.

C-65. C.A. Duncan, M.T. Goodrich, S.G. Kobourov, "Balanced Aspect Ratio Trees: Combining the Benefits of k-D Trees and Octrees," *10th ACM-SIAM Symp. on Discrete Algorithms* (SODA), 1999, 300–309. (Preliminary version of J-41.)

C-66. R.S. Baker, M. Boilen, M.T. Goodrich, R. Tamassia, and B.A. Stibel, "Testers and Visualizers for Teaching Data Structures," *30th ACM SIGCSE Technical Symp. on Computer Science Education*, 1999, 261–265.

C-67. M.T. Goodrich and R. Tamassia, "Using Randomization in the Teaching of Data Structures and Algorithms," *30th ACM SIGCSE Technical Symp. on Computer Science Education*, 1999, 53–57. (Preliminary version of Ch-5.)

C-68. G. Barequet, C. Duncan, M.T. Goodrich, S. Kumar, M. Pop, "Efficient Perspective-Accurate Silhouette Computation," *15th ACM Symp. on Computational Geometry* (SoCG), 1999, 417–418, and the *Video Review for the 15th ACM Symp. on Computational Geometry* (SoCG).

C-69. C.C. Cheng, C.A. Duncan, M.T. Goodrich, and S.G. Kobourov, "Drawing Planar Graphs with Circular Arcs," *7th Int. Symp. on Graph Drawing* (GD), Lecture Notes in Computer Science 1731, Springer-Verlag, 1999, 117–126. (Preliminary version of J-43.)

C-70. C.A. Duncan, M.T. Goodrich, and S.G. Kobourov, "Planarity-Preserving Clustering and Embedding for Large Planar Graphs," *7th Int. Symp. on Graph Drawing* (GD), Lecture Notes in Computer Science 1731, Springer-Verlag, 1999, 186–196. (Preliminary version of J-48.)

C-71. M.T. Goodrich and J.G. Kloss II, "Tiered Vectors: Efficient Dynamic Arrays for Rank-Based Sequences," *1999 Workshop on Algorithms and Data Structures* (WADS), Lecture Notes in Computer Science 1663, Springer-Verlag, 1999, 205–216.

C-72. M.T. Goodrich, "Competitive Tree-Structured Dictionaries," *11th ACM-SIAM Symp. on Discrete Algorithms* (SODA), 2000, 494–495.

C-73. N.M. Amato, M.T. Goodrich, and E.A. Ramos, "Computing the Arrangement of Curve Segments: Divide-and-Conquer Algorithms via Sampling," *11th ACM-SIAM Symp. on Discrete Algorithms* (SODA), 2000, 705–706.

C-74. S. Bridgeman, M.T. Goodrich, S.G. Kobourov, and R. Tamassia, "PILOT: An Interactive Tool for Learning and Grading," *31st ACM SIGCSE Technical Symp. on Computer Science Education*, 2000, 139–143.

C-75. S. Bridgeman, M.T. Goodrich, S.G. Kobourov, and R. Tamassia, "SAIL: A System for Generating, Archiving, and Retrieving Specialized Assignments in LaTeX," *31st ACM SIGCSE Technical Symp. on Computer Science Education*, 2000, 300–304.

C-76. N.M. Amato, M.T. Goodrich, and E.A. Ramos, "Linear-Time Triangulation of a Simple Polygon Made Easier Via Randomization," *16th ACM Symp. on Computational Geometry* (SoCG), 2000, 201-212. (Preliminary version of J-44.)

C-77. A.L. Buchsbaum, M.T. Goodrich, and J.R. Westbrook, "Range Searching Over Tree Cross Products," *8th European Symp. on Algorithms* (ESA), Lecture Notes in Computer Science 1879, Springer-Verlag, 2000, 120–131.

C-78. C.A. Duncan, M.T. Dickerson, and M.T. Goodrich, "k-D Trees are Better When Cut on

the Longest Side," *8th European Symp. on Algorithms* (ESA), Lecture Notes in Computer Science 1879, Springer-Verlag, 2000, 179–190.

C-79. P. Gajer, M.T. Goodrich, and S.G. Kobourov, "A Fast Multi-Dimensional Algorithm for Drawing Large Graphs," *8th Int. Symp. on Graph Drawing* (GD), Lecture Notes in Computer Science 1984, Springer-Verlag, 2001, 211–221. (Preliminary version of J-52.)

C-80. G. Ateniese, B. de Medeiros, and M.T. Goodrich, "TRICERT: A Distributed Certified E-mail Scheme," *Network and Distributed Systems Security Symp.* (NDSS), 2001, 47–56.

C-81. M.T. Goodrich and R. Tamassia, "Teaching Internet Algorithmics," *32nd ACM SIGCSE Technical Symp. on Computer Science Education*, 2001, 129–133.

C-82. M. Pop, G. Barequet, C.A. Duncan, M.T. Goodrich, W. Hwang, and S. Kumar, "Efficient Perspective-Accurate Silhouette Computation and Applications," *17th ACM Symp. on Computational Geometry* (SoCG), 2001, 60–68.

C-83. M.T. Goodrich and R. Tamassia, "Implementation of an Authenticated Dictionary with Skip Lists and Commutative Hashing," *DARPA Information Survivability Conf. & Exposition II* (DISCEX), IEEE Press, 2001, 68–82.

C-84. A. Bagchi, A. Chaudhary, R. Garg, M.T. Goodrich, and V. Kumar, "Seller-Focused Algorithms for Online Auctioning," *2001 Workshop on Algorithms and Data Structures* (WADS), Lecture Notes in Computer Science 2125, Springer-Verlag, 2001, 135–147.

C-85. A. Anagnostopoulos, M.T. Goodrich, R. Tamassia, "Persistent Authenticated Dictionaries and Their Applications," *Information Security Conf.* (ISC), Lecture Notes in Computer Science 2200, Springer-Verlag, 2001, 379–393.

C-86. M.T. Goodrich, R. Tamassia, and J. Hasic, "An Efficient Dynamic and Distributed Cryptographic Accumulator," *5th Information Security Conf.* (ISC), Lecture Notes in Computer Science 2433, Springer-Verlag, 2002, 372–388.

C-87. A.L. Buchsbaum and M.T. Goodrich, "Three-Dimensional Layers of Maxima," *10th European Symp. on Algorithms* (ESA), Lecture Notes in Computer Science 2461, Springer-Verlag, 2002, 257–267. (Preliminary version of J-49.)

C-88. A. Bagchi, A.L. Buchsbaum, and M.T. Goodrich, "Biased Skip Lists," *13th Int. Symp. on Algorithms and Computation* (ISAAC), Lecture Notes in Computer Science 2518, Springer-Verlag, 2002, 1–13. (Preliminary version of J-54.)

C-89. M.T. Goodrich, "Efficient Packet Marking for Large-Scale IP Traceback," *9th ACM Conf. on Computer and Communications Security* (CCS), 2002, 117–126. (Preliminary version of J-61.)

C-90. M.T. Goodrich, R. Tamassia, N. Triandopoulos, and R. Cohen, "Authenticated Data Structures for Graph and Geometric Searching," *RSA Conf.—Cryptographers' Track* (CT-RSA), Lecture Notes in Computer Science 2612, Springer-Verlag, 2003, 295–313. (Preliminary version of J-71.)

C-91. M.T. Goodrich, M. Shin, R. Tamassia, and W.H. Winsborough, "Authenticated Dictionaries for Fresh Attribute Credentials," *1st Int. Conf. on Trust Management* (iTrust), Lecture Notes in Computer Science 2692, Springer-Verlag, 2003, 332–347.

C-92. G. Barequet, M.T. Goodrich, A. Levi-Steiner, and D. Steiner, "Straight-Skeleton Based Contour Interpolation," *14th ACM-SIAM Symp. on Discrete Algorithms* (SODA), 2003, 119–127. (Preliminary version of J-51.)

C-93. G. Barequet, M.T. Goodrich, and C. Riley, "Drawing Graphs with Large Vertices and Thick Edges," *2003 Workshop and Data Structures and Algorithms* (WADS), Lecture Notes in Computer Science 2748, Springer-Verlag, 2003, 281–293. (Preliminary version of J-50.)

C-94. A. Bagchi, A. Chaudhary, M.T. Goodrich, and S. Xu, "Constructing Disjoint Paths for

Secure Communication," *17th Int. Symp. on Distributed Computing* (DISC), Lecture Notes in Computer Science 2848, Springer-Verlag, 2003, 181–195.

C-95. M. Dickerson, D. Eppstein, M.T. Goodrich, J. Meng, "Confluent Drawings: Visualizing Non-planar Diagrams in a Planar Way," *11th Int. Symp. on Graph Drawing* (GD), Lecture Notes in Computer Science 2912, Springer-Verlag, 2003, 1–12. (Preliminary version of J-55.)

C-96. F. Brandenberg, D. Eppstein, M.T. Goodrich, S. Kobourov, G. Liotta, P. Mutzel, "Selected Open Problems in Graph Drawing," *11th Int. Symp. on Graph Drawing* (GD), Lecture Notes in Computer Science 2912, Springer-Verlag, 2003, 515–539.

C-97. A. Bagchi, A. Chaudhary, D. Eppstein, and M.T. Goodrich, "Deterministic Sampling and Range Counting in Geometric Data Streams," *20th ACM Symp. on Computational Geometry* (SoCG), 144–151, 2004. (Preliminary version of J-58.)

C-98. M.T. Goodrich, J.Z. Sun, and R. Tamassia, "Efficient Tree-Based Revocation in Groups of Low-State Devices," *Advances in Cryptology* (CRYPTO), Springer, Lecture Notes in Computer Science 3152, 511–527, 2004.

C-99. D. Eppstein, M.T. Goodrich, and J.Y. Meng, "Confluent Layered Drawings," *12th Int. Symp. on Graph Drawing* (GD), Springer, Lecture Notes in Computer Science 3383, 184–194, 2004. (Preliminary version of J-57.)

C-100. M.J. Atallah, K.B. Frikken, M.T. Goodrich, and R. Tamassia, "Secure Biometric Authentication for Weak Computational Devices," *9th Int. Conf. on Financial Cryptograpy and Data Security*, Springer, Lecture Notes in Computer Science 3570, 357–371, 2005.

C-101. M.T. Goodrich, "Leap-Frog Packet Linking and Diverse Key Distributions for Improved Integrity in Network Broadcasts," *IEEE Symp. on Security and Privacy* (S&P), 196–207, 2005.

C-102. D. Eppstein, M.T. Goodrich, and J.Z. Sun, "The Skip Quadtree: A Simple Dynamic Data Structure for Multidimensional Data," *21st ACM Symp. on Computational Geometry* (SoCG), 296–305, 2005. (Preliminary version of J-60.)

C-103. M.J. Atallah, M.T. Goodrich, and R. Tamassia, "Indexing Information for Data Forensics," *3rd Applied Cryptography and Network Security Conf.* (ACNS), Lecture Notes in Computer Science 3531, Springer, 206–221, 2005.

C-104. W. Du and M.T. Goodrich, "Searching for High-Value Rare Events with Uncheatable Grid Computing," *3rd Applied Cryptography and Network Security Conf.* (ACNS), Lecture Notes in Computer Science 3531, Springer, 122–137, 2005.

C-105. L. Arge, D. Eppstein, and M.T. Goodrich, "Skip-Webs: Efficient Distributed Data Structures for Multi-Dimensional Data Sets," *24th ACM Symp. on Principles of Distributed Computing* (PODC), 2005.

C-106. D. Eppstein, M.T. Goodrich, and D. Hirschberg, "Improved Combinatorial Group Testing for Real-World Problem Sizes," *Workshop on Algorithms and Data Structures* (WADS), Lecture Notes in Computer Science 3608, Springer, 86–98, 2005. (Preliminary version of J-59.)

C-107. A. Chaudhary and M.T. Goodrich, "Balanced Aspect Ratio Trees Revisited," *Workshop on Algorithms and Data Structures* (WADS), Lecture Notes in Computer Science 3608, Springer, 73–85, 2005.

C-108. M.T. Goodrich, R. Tamassia, and D. Yao, "Accredited DomainKeys: A Service Architecture for Improved Email Validation," *2nd Conf. on Email and Anti-Spam* (CEAS), 1–8, 2005.

C-109. M.T. Goodrich, G.S. Lueker, and J.Z. Sun, "C-Planarity of Extrovert Clustered Graphs," *13th Int. Symp. Graph Drawing* (GD), 211–222, 2005.

C-110. D. Eppstein, M.T. Goodrich, J.Y. Meng, "Delta-Confluent Drawings," *13th Int. Symp.*

Graph Drawing (GD), 165–176, 2005.

C-111. M.T. Goodrich, M.J. Nelson, and J.Z. Sun, "The Rainbow Skip Graph: A Fault-Tolerant Constant-Degree Distributed Data Structure," *17th ACM-SIAM Symp. on Discrete Algorithms* (SODA), 384–393, 2006.

C-112. M.T. Goodrich, M. Sirivianos, J. Solis, G. Tsudik, E. Uzun, "Loud And Clear: Human-Verifiable Authentication Based on Audio," *26th IEEE Int. Conf. on Distributed Computing Systems* (ICDCS), 1–8, 2006. (Preliminary version of J-66.)

C-113. M.T. Goodrich, R. Tamassia, and D. Yao, "Notarized Federated Identity Management for Web Services," *20th IFIP WG Working Conf. on Data and Application Security* (DBSec), Springer, Lecture Notes in Computer Science, Vol. 4127, 133–147, 2006. (Preliminary version of J-63.)

C-114. M.T. Goodrich and D.S. Hirschberg, "Efficient Parallel Algorithms for Dead Sensor Diagnosis and Multiple Access Channels," *18th ACM Symp. on Parallelism in Algorithms and Architectures* (SPAA), 118–127, 2006. (Preliminary version of J-62.)

C-115. Y. Cho, L. Bao, and M.T. Goodrich, "LAAC: A Location-Aware Access Control Protocol," *2006 3rd Annual Int. Conf. on Mobile and Ubiquitous Systems - Workshop on Ubiquitous Access Control* (IWUAC), 1–7, 2006.

C-116. M.B. Dillencourt, D. Eppstein, and M.T. Goodrich, "Choosing Colors for Geometric Graphs via Color Space Embeddings," *14th Int. Symp. Graph Drawing* (GD), Lecture Notes in Computer Science, Vol. 4372, Springer, 294–305, 2006.

C-117. D. Eppstein, M.T. Goodrich, and N. Sitchinava, "Guard Placement for Wireless Localization," *23rd ACM Symp. on Computational Geometry* (SoCG), 27–36, 2007.

C-118. M.T. Goodrich, C. Papamanthou, and R. Tamassia, "On the Cost of Persistence and Authentication in Skip Lists," *6th Workshop on Experimental Algorithms* (WEA), LNCS 4525, 94–107, 2007.

C-119. M.J. Atallah, M. Blanton, M.T. Goodrich, and S. Polu, "Discrepancy-Sensitive Dynamic Fractional Cascading, Dominated Maxima Searching, and 2-d Nearest Neighbors in Any Minkowski Metric," *Workshop on Algorithms and Data Structures* (WADS), LNCS, Vol. 4619, Springer, 114–126, 2007.

C-120. D. Eppstein and M.T. Goodrich, "Space-Efficient Straggler Identification in Round-Trip Data Streams via Newton's Identities and Invertible Bloom Filters," *Workshop on Algorithms and Data Structures* (WADS), LNCS, Vol. 4619, Springer, 2007, 638–649. (Preliminary version of J-72.)

C-121. M.T. Goodrich and J.Z. Sun, "Checking Value-Sensitive Data Structures in Sublinear Space," *18th Int. Symp. on Algorithms and Computation* (ISAAC), LNCS, vol. 4835, Springer, 2007, 353–364.

C-122. M.T. Goodrich, R. Tamassia, and N. Triandopoulos, "Super-Efficient Verification of Dynamic Outsourced Databases," *RSA Conf.—Cryptographers' Track* (CT-RSA), LNCS, vol. 4964, Springer, 2008, 407–424.

C-123. D. Eppstein, M.T. Goodrich, E. Kim, and R. Tamstorf, "Approximate Topological Matching of Quadrilateral Meshes," *IEEE Int. Conf. on Shape Modeling and Applications* (SMI), 2008, 83–92. (Preliminary version of J-68.)

C-124. G. Barequet, D. Eppstein, M.T. Goodrich, and A. Waxman, "Straight Skeletons of Three-Dimensional Polyhedra," *16th European Symp. on Algorithms* (ESA), LNCS, vol. 5193, 2008, 148–160.

C-125. M.T. Goodrich, C. Papamanthou, R. Tamassia, and N. Triandopoulos, "Athos: Efficient

Authentication of Outsourced File Systems," *11th Information Security Conf.* (ISC), LNCS, vol. 5222, 2008, 80–96.

C-126. L. Arge, M.T. Goodrich, M. Nelson, and N. Sitchinava, "Fundamental Parallel Algorithms for Private-Cache Chip Multiprocessors," *20th ACM Symp. on Parallelism in Algorithms and Architectures* (SPAA), 2008, 197–206.

C-127. D. Eppstein and M.T. Goodrich, "Succinct Greedy Graph Drawing in the Hyperbolic Plane," *16th Int. Symp. on Graph Drawing* (GD), LNCS, vol. 5417, Springer, 2008, 14–25. (Preliminary version of J-69.)

C-128. D. Eppstein and M.T. Goodrich, "Studying (Non-Planar) Road Networks Through an Algorithmic Lens," *16th ACM SIGSPATIAL Int. Conf. on Adv. in Geographic Information Systems* (GIS), 2008, 125–134. **Best Paper Award**.

C-129. M. Dickerson and M.T. Goodrich, "Two-Site Voronoi Diagrams in Geographic Networks," *16th ACM SIGSPATIAL Int. Conf. on Adv. in Geographic Information Systems* (GIS), 2008, 439–442.

C-130. D. Eppstein, M.T. Goodrich, and D. Strash, "Linear-Time Algorithms for Geometric Graphs with Sublinearly Many Crossings," *20th ACM-SIAM Symp. on Discrete Algorithms* (SODA), 2009, 150–159. (Preliminary version of J-70.)

C-131. M.T. Goodrich, "The Mastermind Attack on Genomic Data," *30th IEEE Symp. on Security and Privacy* (S&P), 2009, 204–218. (Preliminary version of J-80.)

C-132. W. Du, D. Eppstein, M.T. Goodrich, and G.S. Lueker, "On the Approximability of Geometric and Geographic Generalization and the Min-Max Bin Covering Problem," *Algorithms and Data Structures Symp.* (WADS), LNCS, vol. 5664, Springer, 2009, 242–253.

C-133. M.T. Goodrich, R. Tamassia, and N. Triandopoulos, J.Z. Sun, "Reliable Resource Searching in P2P Networks," *5th Int. ICST Conf. on Security and Privacy in Communication Networks* (SecureComm), Lecture Notes of ICST, vol. 19, Springer, 2009, 437–447.

C-134. C.A. Duncan, M.T. Goodrich, S.G. Kobourov, "Planar Drawings of Higher-Genus Graphs," *17th Int. Symp. on Graph Drawing* (GD), LNCS, Springer, vol. 5849, 2009, 45–56. (Preliminary version of J-73.)

C-135. D. Eppstein, M.T. Goodrich, L. Trott, "Going Off-road: Transversal Complexity in Road Networks," *17th ACM SIGSPATIAL Int. Conf. on Adv. in Geographic Information Systems* (GIS), 2009, 23–32.

C-136. M.T. Goodrich and Darren Strash, "Succinct Greedy Geometric Routing in the Euclidean Plane," *20th Int. Symp. on Algorithms and Computation* (ISAAC), LNCS, vol. 5878, Springer, 2009, 781–791.

C-137. M.T. Goodrich, "Randomized Shellsort: A Simple Oblivious Sorting Algorithm," *21st ACM-SIAM Symp. on Discrete Algorithms* (SODA), 2010, 1262–1277. (Preliminary version of J-75.)

C-138. L. Arge, M.T. Goodrich, and N. Sitchinava, "Parallel External Memory Graph Algorithms," *24th IEEE Int. Parallel & Distributed Processing Symp.* (IPDPS), 2010, 1–11.

C-139. G. Wang, T. Luo, M.T. Goodrich, W. Du, and Z. Zhu, "Bureaucratic Protocols for Secure Two-Party Sorting, Selection, and Permuting," *5th ACM Symp. on Information, Computer and Communications Security*, 2010, 226–237.

C-140. M.T. Dickerson, M.T. Goodrich, and T.D. Dickerson, "Round-Trip Voronoi Diagrams and Doubling Density in Geographic Networks," *7th Int. Symp. on Voronoi Diagrams in Science and Engineering* (ISVD), IEEE Press, 132–141, 2010. (Preliminary version of J-74.)

C-141. M.T. Dickerson, D. Eppstein, and M.T. Goodrich, "Cloning Voronoi Diagrams via

Retroactive Data Structures," *18th European Symp. on Algorithms* (ESA), LNCS, vol. 6346, 2010, 362–373.

C-142. C.A. Duncan, D. Eppstein, M.T. Goodrich, S. Kobourov, and M. Nöllenburg, "Lombardi Drawings of Graphs," *18th Int. Symp. on Graph Drawing* (GD), LNCS, vol. 6502, 2010, 195–207. (Preliminary version of J-76.)

C-143. E. Wolf-Chambers, D. Eppstein, M.T. Goodrich, and M. Löffler, "Drawing Graphs in the Plane with a Prescribed Outer Face and Polynomial Area," *18th Int. Symp. on Graph Drawing* (GD), LNCS, vol. 6502, 2010, 129–140. (Preliminary version of J-77.)

C-144. C.A. Duncan, D. Eppstein, M.T. Goodrich, S. Kobourov, and M. Nöllenburg, "Drawing Trees with Perfect Angular Resolution and Polynomial Area," *18th Int. Symp. on Graph Drawing* (GD), LNCS, vol. 6502, 2010, 183–194. (Preliminary version of J-82.)

C-145. A.U. Asuncion and M.T. Goodrich, "Turning Privacy Leaks into Floods: Surreptitious Discovery of Social Network Friendships and Other Sensitive Binary Attribute Vectors," *Workshop on Privacy in the Electronic Society* (WPES), held in conjunction with the 17th ACM Conf. on Computer and Communications Security (CCS), 2010, 21–30. (Preliminary version of J-81.)

C-146. D. Eppstein, M.T. Goodrich, D. Strash, and L. Trott, "Extended Dynamic Subgraph Statistics Using h-Index Parameterized Data Structures," *4th Annual Int. Conf. on Combinatorial Optimization and Applications* (COCOA), LNCS, vol. 6508, 2010, 128–141. (Preliminary version of J-79.)

C-147. M.T. Goodrich and D. Strash, "Priority Range Trees," *21st Int. Symp. on Algorithms and Computation* (ISAAC), LNCS, vol. 6506, 2010, 97–108.

C-148. D. Eppstein, M.T. Goodrich, R. Tamassia, "Privacy-Preserving Data-Oblivious Geometric Algorithms for Geographic Data," *18th ACM SIGSPATIAL Int. Conf. on Adv. in Geographic Information Systems* (GIS), 2010, 13–22.

C-149. M.T. Goodrich, "Spin-the-bottle Sort and Annealing Sort: Oblivious Sorting via Round-robin Random Comparisons," *8th Workshop on Analytic Algorithmics and Combinatorics* (ANALCO), in conjunction with the ACM-SIAM Symp. on Discrete Algorithms (SODA), 2011. (Preliminary version of J-85.)

C-150. M.T. Goodrich and F. Kerschbaum, "Privacy-Enhanced Reputation-Feedback Methods to Reduce Feedback Extortion in Online Auctions," *ACM Conf. on Data and Application Security and Privacy (CODASPY)*, 2011, 273–282.

C-151. M.T. Goodrich, "Data-Oblivious External-Memory Algorithms for the Compaction, Selection, and Sorting of Outsourced Data," *23rd ACM Symp. on Parallelism in Algorithms and Architectures* (SPAA), 2011, 379–388.

C-152. M.T. Goodrich and M. Mitzenmacher, "Brief Announcement: Large-Scale Multimaps," *23rd ACM Symp. on Parallelism in Algorithms and Architectures* (SPAA), 2011, 259–260.

C-153. D. Eppstein, M.T. Goodrich, F. Uyeda, and G. Varghese, "What's the Difference? Efficient Set Synchronization without Prior Context," *SIGCOMM* 218–229, 2011.

C-154. D. Eppstein, M.T. Goodrich, and M. Löffler, "Tracking Moving Objects with Few Handovers," *Algorithms and Data Structures Symp.* (WADS), 362–373, LNCS, vol. 6844, 2011.

C-155. M.T. Goodrich and M. Mitzenmacher, "Privacy-Preserving Access of Outsourced Data via Oblivious RAM Simulation," *38th Int. Colloquium on Automata, Languages and Programming (ICALP)*, LNCS, vol. 6756, 2011, 576–587.

C-156. M.T. Goodrich and P. Pszona, "External-Memory Network Analysis Algorithms for

Naturally Sparse Graphs," *European Symp. on Algorithms (ESA)*, LNCS, vol. 6942, 664–676, 2011.

C-157. C. Duncan, D. Eppstein, M.T. Goodrich, S.G. Kobourov and M. Löffler, "Planar and Poly-Arc Lombardi Drawings," *Int. Symp. Graph Drawing (GD)*, LNCS, vol. 7034, 308–319, 2011. (Preliminary version of J-87.)

C-158. R. Chernobelskiy, K. Cunningham, M.T. Goodrich, S.G. Kobourov and L. Trott, "Force-Directed Lombardi-Style Graph Drawing," *Int. Symp. Graph Drawing (GD)*, LNCS, vol. 7034, 320–331, 2011.

C-159. M.T. Goodrich and M. Mitzenmacher, "Invertible Bloom Lookup Tables," *49th Allerton Conf. on Communication, Control, and Computing*, IEEE Press, **invited paper**, 2011.

C-160. M.T. Goodrich, M. Mitzenmacher, O. Ohrimenko, and R. Tamassia, "Oblivious RAM Simulation with Efficient Worst-Case Access Overhead," *ACM Cloud Computing Security Workshop (CCSW)*, in conjunction with the 17th ACM Conf. on Computer and Communications Security (CCS), 95–100, 2011.

C-161. M.T. Goodrich and J.A. Simons, "Fully Retroactive Approximate Range and Nearest Neighbor Searching," *22nd Int. Symp. on Algorithms and Computation (ISAAC)*, Springer, LNCS, vol. 7074, 292–301, 2011.

C-162. E. Angelino, M.T. Goodrich, M. Mitzenmacher and J. Thaler, "External Memory Multimaps," *22nd Int. Symp. on Algorithms and Computation (ISAAC)*, Springer, LNCS, vol. 7074, 384–394, 2011. (Preliminary version of J-83.)

C-163. M.T. Goodrich, N. Sitchinava, and Q. Zhang, "Sorting, Searching, and Simulation in the MapReduce Framework," *22nd Int. Symp. on Algorithms and Computation (ISAAC)*, Springer, LNCS, vol. 7074, 374–383, 2011.

C-164. D. Eppstein, M.T. Goodrich, M. Löffler, D. Strash and L. Trott, "Category-Based Routing in Social Networks: Membership Dimension and the Small-World Phenomenon," *IEEE Int. Conf. on Computational Aspects of Social Networks* (CASoN), 102–107, 2011. (Preliminary version of J-84.)

C-165. M.T. Goodrich, O. Ohrimenko, M. Mitzenmacher, and R. Tamassia, "Privacy-Preserving Group Data Access via Stateless Oblivious RAM Simulation," *23rd ACM-SIAM Symp. on Discrete Algorithms* (SODA), 157–167, 2012.

C-166. M.T. Goodrich, O. Ohrimenko, M. Mitzenmacher, and R. Tamassia, "Practical Oblivious Storage," *2nd ACM Conf. on Data and Application Security and Privacy* (CODASPY). 13–24, 2012.

C-167. M.T. Goodrich and M. Mitzenmacher, "Anonymous Card Shuffling and its Applications to Parallel Mixnets," *39th Int. Colloquium on Automata, Languages and Programming* (ICALP), Springer, LNCS, vol. 6756, 576–587, 2012.

C-168. M.T. Goodrich, O. Ohrimenko, and R. Tamassia, "Graph Drawing in the Cloud: Privately Visualizing Relational Data using Small Working Storage," *20th Int. Symp. on Graph Drawing* (GD), Springer, LNCS, vol. 7704, 43–54, 2012.

C-169. F.J. Brandenburg, D. Eppstein, A. Gleissner, M.T. Goodrich, K. Hanauer, and J. Reislhuber, "On the Density of Maximal 1-Planar Graphs," *20th Int. Symp. on Graph Drawing* (GD), Springer, LNCS, vol. 7704, 327–338, 2012.

C-170. M.J. Bannister, D. Eppstein, M.T. Goodrich, and L. Trott, "Force-Directed Graph Drawing Using Social Gravity and Scaling," *20th Int. Symp. on Graph Drawing* (GD), Springer, LNCS, vol. 7704, 414–425, 2012.

C-171. M.T. Goodrich and J.A. Simons, "More Graph Drawing in the Cloud: Data-Oblivious

st-Numbering, Visibility Representations, and Orthogonal Drawing of Biconnected Planar Graphs," *20th Int. Symp. on Graph Drawing* (GD), Springer, LNCS, vol. 7704, 569–570, 2012.

C-172. M.T. Goodrich, D.S. Hirschberg, M. Mitzenmacher, and J. Thaler, "Cache-Oblivious Dictionaries and Multimaps with Negligible Failure Probability," *Mediterranean Conf. on Algorithms* (MedAlg), Springer, LNCS, vol. 7659, 203–218, 2012.

C-173. D. Eppstein, M.T. Goodrich, and D.S. Hirschberg, "Combinatorial Pair Testing: Distinguishing Workers from Slackers," *Algorithms and Data Structures Symp.* (WADS), Springer, LNCS, vol. 8037, 316–327, 2013.

C-174. D. Eppstein, M.T. Goodrich, and J.A. Simons, "Set-Difference Range Queries," *25th Canadian Conf. on Computational Geometry* (CCCG), 2013, http://www.cccg.ca/proceedings/2013/.

C-175. M.T. Goodrich and P. Pszona, "Cole's Parametric Search Technique Made Practical," *25th Canadian Conf. on Computational Geometry* (CCCG), 2013, http://www.cccg.ca/proceedings/2013/.

C-176. L. Arge, M.T. Goodrich, F. van Walderveen, "Computing Betweenness Centrality in External Memory," *IEEE Int. Conf. on Big Data* (BigData), 368–375, 2013.

C-177. M.T. Goodrich and P. Pszona, "Achieving Good Angular Resolution in 3D Arc Diagrams," *21st Int. Symp. Graph Drawing* (GD), Springer, LNCS, vol. 8242, 161–172, 2013.

C-178. M.T. Goodrich and P. Pszona, "Streamed Graph Drawing and the File Maintenance Problem," *21st Int. Symp. Graph Drawing* (GD), Springer, LNCS, vol. 8242, 256–267, 2013.

C-179. M.T. Goodrich, "Zig-zag Sort: A Simple Deterministic Data-Oblivious Sorting Algorithm Running in $O(n \log n)$ Time," *46th ACM Symp. on Theory of Computing* (STOC), 684–693, 2014.

C-180. D. Eppstein, M.T. Goodrich, M. Mitzenmacher, and P. Pszona, "Wear Minimization for Cuckoo Hashing: How Not to Throw a Lot of Eggs into One Basket," *Symp. on Experimental Algorithms* (SEA), Springer, LNCS, vol. 8504, 162–173, 2014.

C-181. O. Ohrimenko, M.T. Goodrich, and R. Tamassia, an E. Upfal, "The Melbourne Shuffle: Improving Oblivious Storage in the Cloud," *41st Int. Colloq. on Automata, Languages, and Programming* (ICALP), Springer, LNCS, vol. 8573, 556–567, 2014.

C-182. M.J. Bannister, W.E. Devanny, M.T. Goodrich, J.A. Simons, and Lowell Trott, "Windows into Geometric Events: Data Structures for Time-Windowed Querying of Temporal Point Sets," *26th Canadian Conf. on Computational Geometry* (CCCG), 2014.

C-183. M.J. Bannister, W.E. Devanny, D. Eppstein and M.T. Goodrich, "The Galois Complexity of Graph Drawing: Why Numerical Solutions are Ubiquitous for Force-Directed, Spectral, and Circle Packing Drawings," *22nd Int. Symp. Graph Drawing* (GD), Springer, LNCS, vol. 8871, 149–161, 2014. (Preliminary version of J-86.)

C-184. M.J. Alam, D. Eppstein, M.T. Goodrich, S. Kobourov and S. Pupyrev, "Balanced Circle Packings for Planar Graphs," *22nd Int. Symp. Graph Drawing* (GD), Springer, LNCS, vol. 8871, 125–136, 2014.

C-185. M. Bannister, M.T. Goodrich, and P. Sampson, "Force-Directed 3D Arc Diagrams," *22nd Int. Symp. Graph Drawing* (GD), Springer, LNCS, vol. 8871, 521–522, 2014.

C-186. M.T. Goodrich and P. Pszona, "Two-Phase Bicriterion Search for Finding Fast and Efficient Electric Vehicle Routes," *22nd ACM SIGSPATIAL Int. Conf. on Adv. Geographic Information Systems* (GIS), 193–202, 2014.

C-187. M.T. Goodrich and J. Simons, "Data-Oblivious Graph Algorithms in Outsourced External

Memory," *8th Int. Conf. on Combinatorial Optimization and Applications* (COCOA), LNCS, Vol. 8881, 241–257, 2014.

C-188. M.T. Goodrich, T. Johnson, M. Torres, "Knuthian Drawings of Series-Parallel Flowcharts," *23rd Int. Symp. on Graph Drawing and Network Visualization* (GD), Springer, LNCS, vol. 9411, 556–557, 2015. (See also http://arxiv.org/abs/1508.03931.)

C-189. M.T. Goodrich and A. Eldawy, "Parallel Algorithms for Summing Floating-Point Numbers," *28th ACM Symp. on Parallel Algorithms and Architectures* (SPAA), 13–22, 2016.

C-190. W.E. Devanny, M.T. Goodrich, and K. Jetviroj, "Parallel Equivalence Class Sorting: Algorithms, Lower Bounds, and Distribution-Based Analysis," *28th ACM Symp. on Parallel Algorithms and Architectures* (SPAA), 265–274, 2016.

C-191. D. Eppstein, M.T. Goodrich, J. Lam, N. Mamano, M. Mitzenmacher, and M. Torres, "Models and Algorithms for Graph Watermarking," *19th Information Security Conf.* (ISC), 283–301, 2016. **Best Student Paper Award**.

C-192. E. Ghosh, M.T. Goodrich, O. Ohrimenko, R. Tamassia, "Verifiable Zero-Knowledge Order Queries and Updates for Fully Dynamic Lists and Trees," *10th Conf. on Security and Cryptography for Networks* (SCN), 216–236, 2016.

C-193. M.T. Goodrich, E. Kornaropoulos, M. Mitzenmacher, R. Tamassia, "More Practical and Secure History-Independent Hash Tables," *21st European Symp. on Research in Computer Security* (ESORICS), 20-38, 2016.

C-194. J.J. Besa Vial, W.E. Devanny, D. Eppstein, and M.T. Goodrich, "Scheduling Autonomous Vehicle Platoons Through an Unregulated Intersection," *2016 Workshop on Algorithmic Approaches for Transportation Modeling, Optimization, and Systems* (ATMOS), 5:1–5:14.

C-195. M.J. Alam, M.B. Dillencourt, and M.T. Goodrich, "Capturing Lombardi Flow in Orthogonal Drawings by Minimizing the Number of Segments," *24th Int. Symp. on Graph Drawing and Network Visualization* (GD), LNCS, Vol. 9801, 608–610, 2016.

C-196. M.J. Alam, M.T. Goodrich, and T. Johnson, "Sibling-First Recursive Graph Drawing for Java Bytecode," *24th Int. Symp. on Graph Drawing and Network Visualization* (GD), LNCS, Vol. 9801, 611–612, 2016.

C-197. M.T. Goodrich, S. Gupta, and M. Torres, "A Topological Algorithm for Determining How Road Networks Evolve Over Time," *24th ACM SIGSPATIAL Int. Conf. on Advances in Geographic Information Systems* (GIS), 31:1–31:10, 2016.

C-198. M.J. Alam, M.T. Goodrich, and T. Johnson, "J-Viz: Finding Algorithmic Complexity Attacks via Graph Visualization of Java Bytecode," *13th IEEE Symp. on Visualization for Cyber Security* (VizSec), 1–8, 2016.

C-199. M.T. Goodrich, E. Kornaropoulos, M. Mitzenmacher, and R. Tamassia, "Auditable Data Structures," *2nd IEEE European Symp. on Security and Privacy* (EuroS&P), 285–300, 2017.

C-200. D. Eppstein, M.T. Goodrich, M. Mitzenmacher, and M. Torres, "2-3 Cuckoo Filters for Faster Triangle Listing and Set Intersection," *36th ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems* (PODS), 247–260, 2017.

C-201. D. Eppstein, M.T. Goodrich, and N. Mamano, "Algorithms for Stable Matching and Clustering in a Grid," *18th International Workshop on Combinatorial Image Analysis* (IWCIA), 117–131, 2017.

C-202. G. Ateniese, M.T. Goodrich, V. Lekakis, C. Papamanthou, E. Paraskevas, and R. Tamassia, "Accountable Storage," *15th International Conference on Applied Cryptography and Network Security* (ACNS), 623–644, 2017.

C-203. D. Eppstein and M.T. Goodrich, "Brief Announcement: Using Multi-Level Parallelism

and 2-3 Cuckoo Filters for Faster Set Intersection Queries and Sparse Boolean Matrix Multiplication," *29th ACM Symp. on Parallelism in Algorithms and Architectures* (SPAA), 137–139, 2017.

C-204. W.E. Devanny, J. Fineman, M.T. Goodrich, and T. Kopelowitz, "The Online House Numbering Problem: Min-Max Online List Labeling," *25th European Symposium on Algorithms* (ESA), 33:1–33:15, 2017.

C-205. M.T. Goodrich, "Answering Spatial Multiple-Set Intersection Queries Using 2-3 Cuckoo Hash-Filters," *25th ACM SIGSPATIAL Int. Conf. on Advances in Geographic Information Systems* (GIS), 65:1–65:4, 2017.

C-206. D. Eppstein, M.T. Goodrich, D. Korkmaz, and N. Mamano, "Defining Equitable Geographic Districts in Road Networks via Stable Matching," *25th ACM SIGSPATIAL Int. Conf. on Advances in Geographic Information Systems* (GIS), 52:1–52:4, 2017.

C-207. M.T. Goodrich, "BIOS ORAM: Improved Privacy-Preserving Data Access for Parameterized Outsourced Storage,' *ACM Workshop on Privacy in the Electronic Society* (WPES), 41–50, 2017.

C-208. J.J. Besa Vial, W.E. Devanny, D. Eppstein, M.T. Goodrich, and T. Johnson, "Quadratic Time Algorithms Appear to be Optimal for Sorting Evolving Data," *Algorithm Engineering & Experiments* (ALENEX), 87–96, 2018.

C-209. D. Eppstein, M.T. Goodrich, N. Mamano, "Reactive Proximity Data Structures for Graphs," *13th Latin American Theoretical Informatics Symposium* (LATIN), LNCS, Vol. 10807, Springer, 777–789, 2018.

C-210. M.T. Goodrich, "Isogrammic-Fusion ORAM: Improved Statistically Secure Privacy-Preserving Cloud Data Access for Thin Clients," *13th ACM ASIA Conf. on Information, Computer and Communications Security* (ASIACCS), 699–706, 2018.

C-211. J.J. Besa Vial, W.E. Devanny, D. Eppstein, M.T. Goodrich, and T. Johnson, "Optimally Sorting Evolving Data," *45th Int. Colloq. on Automata, Languages, and Programming* (ICALP), 81:1–81:13, 2018.

C-212. G. Barequet, D. Eppstein, M.T. Goodrich, and N. Mamano, "Stable-Matching Voronoi Diagrams: Combinatorial Complexity and Algorithms," *45th Int. Colloq. on Automata, Languages, and Programming* (ICALP), 89:1–89:14, 2018.

C-213. G. Da Lozzo, D. Eppstein, M.T. Goodrich, and S. Gupta, "Subexponential-Time and FPT Algorithms for Embedded Flat Clustered Planarity," *44th Int. Workshop on Graph-Theoretic Concepts in Computer Science* (WG), 111–124, 2018.

C-214. G. Barequet, M. De, and M.T. Goodrich, "Computing Convex-Straight-Skeleton Voronoi Diagrams for Segments and Convex Polygons," *24th International Computing and Combinatorics Conference* (COCOON), 130–142, 2018.

C-215. M.T. Goodrich and T. Johnson, "Low Ply Drawings of Trees and 2-Trees," *30th Canadian Conference on Computational Geometry* (CCCG), 1–9, 2018.

C-216. D. Eppstein, M.T. Goodrich, J. Jorgensen, and M.R. Torres, "Geometric Fingerprint Recognition via Oriented Point-Set Pattern Matching," *30th Canadian Conference on Computational Geometry* (CCCG), 1–16, 2018.

C-217. G. Da Lozzo, D. Eppstein, M.T. Goodrich, and S. Gupta, "C-Planarity Testing of Embedded Clustered Graphs with Bounded Dual Carving-Width," *14th Int. Symp. on Parameterized and Exact Computation* (IPEC), LIPIcs, vol. 148, 9:1–9:17, 2019. **Best Paper Award**.

C-218. J.J. Besa, G. Da Lozzo, and M.T. Goodrich, "Computing k-Modal Embeddings of Planar Digraphs," *European Symposium on Algorithms* (ESA), 19:1–19:16, 2019.

C-219. N. Mamano, A. Efrat, D. Eppstein, D. Frishberg, M.T. Goodrich, S. Kobourov, P. Matias, and V. Polishchuk, "New Applications of Nearest-Neighbor Chains: Euclidean TSP and Motorcycle Graphs," *30th Int. Symp. on Algorithms and Computation* (ISAAC), 51:1–51:21, 2019.

C-220. D. Eppstein, M.T. Goodrich, J.A. Liu, and P.A. Matias, "Tracking Paths in Planar Graphs," *30th Int. Symp. on Algorithms and Computation* (ISAAC), 54:1–54:17, 2019.

C-221. J.J. Besa, M.T. Goodrich, T. Johnson, and M.C. Osegueda, "Minimum-Width Drawings of Phylogenetic Trees," *13th Int. Conf. on Combinatorial Optimization and Applications* (COCOA), LNCS, vol. 11949, 39–55, 2019.

C-222. M.T. Goodrich, Z.M. Liang, and S. Zhao, "Inverse-Rendering Based Analysis of the Fine Illumination Effects in the Salvator Mundi," *ACM SIGGRAPH Art Papers Program, 47th International Conference and Exhibition on Computer Graphics and Interactive Techniques*, 2020.

C-223. R. Afshar, M.T. Goodrich, P. Matias, and M.C. Osegueda, "Reconstructing Binary Trees in Parallel," *32nd ACM Symp. on Parallelism in Algorithms and Architectures* (SPAA), 491–492, 2020.

C-224. R. Afshar, M.T. Goodrich, P. Matias, and M.C. Osegueda, "Reconstructing Biological and Digital Phylogenetic Trees in Parallel," *European Symposium on Algorithms* (ESA), 3:1–3:24, 2020.

C-225. R. Afshar, A. Amir, M.T. Goodrich, and P. Matias, "Adaptive Exact Learning in a Mixed-Up World: Dealing with Periodicity, Errors, and Jumbled-Index Queries in String Reconstruction," *27th International Symposium on String Processing and Information Retrieval* (SPIRE), 2020.

Other Publications:

O-1. M.T. Goodrich, "Guest Editor's Introduction," *International Journal of Computational Geometry & Applications*, **2**(2), 1992, 113–116.

O-2. M.T. Goodrich, "Parallel Algorithms Column 1: Models of Computation," *SIGACT News*, **24**(4), 1993, 16–21.

O-3. M.T. Goodrich, V. Mirelli, M. Orletsky, and J. Salowe, "Decision tree construction in fixed dimensions: Being global is hard but local greed is good," Technical Report TR-95-1, Johns Hopkins University, Department of Computer Science, Baltimore, MD 21218, May 1995.

O-4. R. Tamassia, P.K. Agarwal, N. Amato, D.Z. Chen, D. Dobkin, R.L.S. Drysdale, S. Fortune, M.T. Goodrich, J. Hershberger, J. O'Rourke, F.P. Preparata, J.-R. Sack, S. Suri, I.G. Tollis, J.S. Vitter, and S. Whitesides, "Strategic Directions in Computational Geometry Working Group Report," *ACM Computing Surveys*, **28A**(4), December 1996.

O-5. G.A. Gibson, J.S. Vitter, and J. Wilkes, A. Choudhary, P. Corbett, T.H. Cormen, C.S. Ellis, M.T. Goodrich, P. Highnam, D. Kotz, K. Li, R. Muntz, J. Pasquale, M. Satyanarayanan, D.E. Vengroff, "Report of the Working Group on Storage I/O Issues in Large-Scale Computing," *ACM Computing Surveys*, **28A**(4), December 1996.

O-6. T.H. Cormen and M.T. Goodrich, "A Bridging Model for Parallel Computation, Communication, and I/O," *ACM Computing Surveys*, **28A**(4), December 1996.

O-7. M.T. Goodrich, "Computer Science Issues in the National Virtual Observatory," in *Virtual Observatories of the Future*, ASP Conf. Series, vol. 225, R.J. Brunner, S.G. Djorgovski, and A.S. Szalay, eds., 329–332, 2001.

O-8. M.T. Dickerson and M.T. Goodrich, "Matching Points to a Convex Polygonal Boundary," Proceedings of the 13th Canadian Conf. on Computational Geometry (CCCG'01), 89–92,

2001.

O-9. M.T. Goodrich, "Guest Editor's Foreword," *Algorithmica*, **33**(3), 271, 2002.

O-10. M.T. Goodrich, M. Shin, C.D. Straub, and R. Tamassia, "Distributed Data Authentication (System Demonstration)," *DARPA Information Survivability Conf. and Exposition*, IEEE Press, Volume 2, 58–59, 2003.

O-11. M.T. Goodrich and R. Tamassia, "Efficient and Scalable Infrastructure Support for Dynamic Coalitions," *DARPA Information Survivability Conf. and Exposition*, IEEE Press, Volume 2, 246–251, 2003.

O-12. E. Ghosh, M.T. Goodrich, O. Ohrimenko, and R. Tamassia, "Poster: Zero-Knowledge Authenticated Order Queries and Applications," *IEEE Symp. on Security and Privacy*, 2015. (See also https://eprint.iacr.org/2015/283.)

O-13. F. Bayatbabolghani, M. Blanton, M. Aliasgari, and M.T. Goodrich, "Poster: Secure Computations of Trigonometric and Inverse Trigonometric Functions," *IEEE Symposium on Security and Privacy*, 2017.

PROFESSIONAL SERVICE

Guest Editor:

Int. Journal of Computational Geometry & Applications, **2**(2), 1992
Journal of Computer & System Sciences, **52**(1), 1996
Computational Geometry: Theory and Applications, **12(1–2)**, 1999.
Algorithmica, **33**(3), 2002.

Editorial Board Membership:

Computational Geometry: Theory and Applications, 2006–2015
Journal of Computer & System Sciences, 1994–2011
Journal of Graph Algorithms and Applications, 1996–2011
Int. Journal of Computational Geometry & Applications, 1993–2010
Information Processing Letters, 1995–1997

Journal Advisory Board Membership:

Int. Journal of Computational Geometry & Applications, 2010–
Journal of Graph Algorithms and Applications, 2011–

Program Committee Service:

7th ACM Symp. on Computational Geometry (SoCG), 1991
1991 Workshop on Algorithms and Data Structures (WADS)
8th ACM Symp. on Computational Geometry (SoCG), 1992
25th ACM Symp. on Theory of Computing (STOC), 1993
Chair, 26th ACM Symp. on Theory of Computing (STOC), 1994
11th ACM Symp. on Computational Geometry (SoCG), 1995
DAGS '95 Conf. on Electronic Publishing and the Information Superhighway
1996 SIAM Discrete Mathematics Conference
1997 Workshop on Algorithms and Data Structures (WADS)
International Symposium on Graph Drawing (GD), 1997
1999 Workshop on Algorithms and Data Structures (WADS)
Co-chair, Workshop on Algorithm Engineering and Experimentation (ALENEX), 1999
International Symposium on Graph Drawing (GD), 2000
2000 Workshop on Algorithm Engineering (WAE)
41st IEEE Symp. on Foundations of Computer Science (FOCS), 2000

2001 Workshop on Algorithms and Data Structures (WADS)
International Symposium on Graph Drawing (GD), 2001
Workshop on Algorithm Engineering and Experimentation (ALENEX), 2002
18th ACM Symp. on Computational Geometry (SoCG), 2002
13th ACM-SIAM Symp. on Discrete Algorithms (SODA), 2002
Co-Chair, Graph Drawing 2002
International Symposium on Graph Drawing (GD), 2003
16th ACM-SIAM Symp. on Discrete Algorithms (SODA), 2005
32nd Int. Colloq. on Automata, Languages and Programming (ICALP), 2005
12th Int. Computing and Combinatorics Conference (COCOON), 2006
13th ACM Conf. on Computer and Communication Security (CCS), 2006
15th Annual European Symposium on Algorithms (ESA), 2007
5th International Conference on Applied Cryptography and Network Security (ACNS), 2007
21st IEEE International Parallel & Distributed Processing Symposium (IPDPS), 2007
19th ACM Symposium on Parallelism in Algorithms and Architectures (SPAA), 2007
5th Workshop on Algorithms and Models for the Web-Graph (WAW), 2007
7th International Workshop on Experimental Algorithms (WEA), 2008
Second International Frontiers of Algorithmics Workshop (FAW), 2008
16th ACM SIGSPATIAL Int. Symp. on Adv. in Geographic Information Systems (GIS), 2008
17th ACM SIGSPATIAL Int. Symp. on Adv. in Geographic Information Systems (GIS), 2009
31st IEEE Symposium on Security and Privacy (SSP), 2010
18th Int. Symp. on Graph Drawing (GD), 2010
2011 Workshop on Analytic Algorithmics and Combinatorics (ANALCO)
8th Workshop on Algorithms and Models for the Web Graph (WAW), 2011
19th International Symposium on Graph Drawing (GD), 2011
24th ACM Symposium on Parallelism in Algorithms and Architectures (SPAA), 2012
20th European Symposium on Algorithms (ESA), 2012
2013 IEEE Int. Conf. on Big Data (BigData), 2013
30th IEEE Int. Conf. on Data Engineering (ICDE), 2014
21st ACM Conf. on Computer and Communication Security (CCS), 2014
Sympoisum on Algorithms and Data Structures (WADS), 2015
ACM Cloud Computing Security Workshop (CCSW), 2015
International Symposium on Graph Drawing (GD), 2015
co-chair, 2016 Workshop on Algorithm Engineering and Experiments (ALENEX)
2016 Workshop on Massive Data Algorithmics (MASSIVE)
2016 Int. Symposium on Algorithms and Computation (ISAAC)
29th ACM Symposium on Parallelism in Algorithms and Architectures (SPAA), 2017
25th ACM SIGSPATIAL Int. Conf. on Adv. in Geographic Information Systems (GIS), 2017
26th European Symposium on Algorithms (ESA), 2018
26th ACM SIGSPATIAL Int. Conf. on Adv. in Geographic Information Systems (GIS), 2018
2nd Symposium on Simplicity in Algorithms (SOSA), 2019
1st ACM SIGSPATIAL Int. Workshop on Spatial Gems, 2019
2021 SIAM Applied Computational & Discrete Algorithms (ACDA)

Conference/Workshop Committee Service:

Conference chair, 12th ACM Symposium on Computational Geometry, 1996
Organizer, 1st CGC Workshop on Computational Geometry, 1996
Co-chair, 1999 Dagstuhl Workshop on Computational Geometry, 1999
Conference chair, Graph Drawing, 2002
Co-organizer, Hawaiian Workshop on Parallel Algorithms, 2017, 2019

Steering Committee and Executive Committee Service:

Member at large, ACM SIG on Algorithms & Comp. Theory (SIGACT) Exec. Comm., 1993–97

Member, Exec. comm. for 1996 Federated Computing Research Conference (FCRC)
co-Founder and member, Steering Comm. for Workshop on Algorithm Engineering and Experimentation (ALENEX), 1999–2017 (chair, 2014–16)
co-Chair, Steering Comm. for ACM Symposium on Computational Geometry, 1999–2001
Member, Steering Comm. for Graph Drawing Conference, 2000–03, 2014–16
Conference Chair, ACM SIG on Algorithms & Comp. Theory (SIGACT), 2005–09

Review Panelist:

National Science Foundation, 2000–2017

Postdoctoral Fellows:

1. Timothy Chan, Johns Hopkins, 1996. (Now at Univ. of Illinois)
2. Gill Barequet, Johns Hopkins, 1996-98. (Now at Technion)
3. Pawel Gajer, Johns Hopkins, 2000. (Now at Univ. of Maryland)
4. Amitabh Chaudhary, UC-Irvine, 2002-2004. (Now at U. Chicago)
5. Amitabha Bagchi, UC-Irvine, 2002-2004. (Now at IIT-Dehli)
6. Martin Nollenburg, UC-Irvine, 2010, mentored jointly with David Eppstein. (Now at TU Wien)
7. Maarten Loffler, UC-Irvine, 2010-2011, mentored jointly with David Eppstein. (Now at Utrecht University)
8. Md. Jawaherul Alam, UC-Irvine, 2015-16. (Now at Amazon)
9. Giordano Da Lozzo, UC-Irvine, 2016-2017, mentored jointly with David Eppstein. (Now at "Roma Tre" University)

Ph.D. Students:

1. Mujtaba Ghouse, "Randomized Parallel Computational Geometry in Theory and Practice," Johns Hopkins Univ., May 1993.
2. Paul Tanenbaum, "On Geometric Representations of Partially Ordered Sets," Johns Hopkins Univ., May 1995 (co-advised with Edward Scheinerman).
3. Mark Orletsky, "Practical Methods for Geometric Searching Problems with Experimental Validation," Johns Hopkins Univ., May 1996.
4. Kumar Ramaiyer, "Geometric Data Structures and Applications," Johns Hopkins Univ., Aug. 1996.
5. Christian Duncan, "Balanced Aspect Ratio Trees," Johns Hopkins Univ., Aug. 1999.
6. Christopher Wagner, "Graph Visualization and Network Routing," Johns Hopkins Univ., Oct. 1999 (co-advised with Prof. Lenore Cowen).
7. Stephen Kobourov, "Algorithms for Drawing Large Graphs," Johns Hopkins Univ., May 2000.
8. Amitabha Bagchi, "Efficient Strategies for Topics in Internet Algorithmics," Johns Hopkins Univ., Oct. 2002.
9. Amitabh Chaudhary, "Applied Spatial Data Structures for Large Data Sets," Johns Hopkins Univ., Oct. 2002.
10. Breno De Medeiros, "New Cryptographic Primitives with Applications to Information Privacy and Corporate Confidentiality," Johns Hopkins Univ., May 2004 (co-advised with Giuseppe Ateniese).
11. Jeremy Yu Meng, "Confluent Graph Drawing," UC-Irvine, June 2006.
12. Jonathan Zheng Sun, "Algorithms for Hierarchical Structures, with Applications to Security and Geometry," UC-Irvine, Aug. 2006.

13. Nodari Sitchinava, "Parallel External Memory Model—A Parallel Model for Multi-core Architectures," UC-Irvine, Sep. 2009.
14. Darren Strash, "Algorithms for Sparse Geometric Graphs and Social Networks," UC-Irvine, May 2011 (co-advised with with David Eppstein).
15. Lowell Trott, "Geometric Algorithms for Social Network Analysis," UC-Irvine, May 2013.
16. Joseph Simons, "New Dynamics in Geometric Data Structures," UC-Irvine, May 2014.
17. Pawel Pszona, "Practical Algorithms for Sparse Graphs," UC-Irvine, May 2014.
18. William E. Devanny, "An Assortment of Sorts: Three Modern Variations on the Classic Sorting Problem," UC-Irvine, July 2017 (co-advised with David Eppstein).
19. Siddharth Gupta, "Topological Algorithms for Geographic and Geometric Graphs," UC-Irvine, Aug. 2018 (co-advised with with David Eppstein).
20. Timothy Johnson, "Graph Drawing Representations and Metrics with Applications," UC-Irvine, Aug. 2018.
21. Juan Besa, "Optimization Problems in Directed Graph Visualization," UC-Irvine, Aug. 2019.
22. Nil Mamano Grande, "New Applications of the Nearest-Neighbor Chain Algorithm," UC-Irvine, Sep. 2019 (co-adviced with David Eppstein).

Ph.D. Committee Service:

John Augustine	UC-Irvine	Advancement to candidacy, September 2003
Nikos Triandopoulos	Brown U.	Thesis prelim., February 2004
Einar Mykletun	UC-Irvine	Advancement to candidacy, March 2004
Kartic Subr	UC-Irvine	Advancement to candidacy, September 2004
S. Joshua Swamidass	UC-Irvine	Advancement to candidacy, April 2005
Jeong Hyun Yi	UC-Irvine	Thesis defense, August, 2005
Nodari Sitchinava	UC-Irvine	Advancement to candidacy, chair, December 2005
John Augustine	UC-Irvine	Thesis defense, July 2006
Maithili Narasimha	UC-Irvine	Thesis defense, August, 2006
Josiah Carlson	UC-Irvine	Advancement to candidacy, August 2006
Xiaomin Liu	UC-Irvine	Advancement to candidacy, September 2006
Gabor Madl	UC-Irvine	Advancement to candidacy, September 2006
Nikos Triandopoulos	Brown U.	Thesis defense, September 2006
Rabia Nuray-Turan	UC-Irvine	Advancement to candidacy, May 2007
S. Joshua Swamidass	UC-Irvine	Thesis defense, June 2007
Michael Sirivianos	UC-Irvine	Advancement to candidacy, June 2007
Kevin Wortman	UC-Irvine	Advancement to candidacy, August 2007
Di Ma	UC-Irvine	Advancement to candidacy, December 2007
Josiah Carlson	UC-Irvine	Thesis defense, December 2007
Michael Nelson	UC-Irvine	Advancement to candidacy, chair, March 2008
Minas Gjoka	UC-Irvine	Advancement to candidacy, June 2008
Sara Javanmardi	UC-Irvine	Advancement to candidacy, June 2008
Ali Zandi	UC-Irvine	Advancement to candidacy, September 2008
Jihye Kim	UC-Irvine	Thesis defense, September 2008
Darren Strash	UC-Irvine	Advancement to candidacy, December 2008
Kevin Wortman	UC-Irvine	Topic defense, January 2009
Nodari Sitchinava	UC-Irvine	Topic defense, chair, June 2009
Fabio Soldo	UC-Irvine	Advancement to candidacy, July 2009
Emil De Cristofaro	UC-Irvine	Advancement to candidacy, July 2009
Di Ma	UC-Irvine	Thesis defense, August 2009

Yanbin Lu	UC-Irvine	Advancement to candidacy, December 2009
Anh Le	UC-Irvine	Advancement to candidacy, April 2010
Lowell Trott	UC-Irvine	Advancement to candidacy, June 2010
Xiaomin Liu	UC-Irvine	Thesis defense, August 2010
Josh Olsen	UC-Irvine	Advancement to candidacy, September 2010
Yasser Altowim	UC-Irvine	Advancement to candidacy, December 2010
Angela Wong	UC-Irvine	Advancement to candidacy, May 2011
Joshua Hill	UC-Irvine	Advancement to candidacy, September 2011
Alex Abatzoglou	UC-Irvine	Advancement to candidacy, September 2011
Michael Wolfe	UC-Irvine	Masters Thesis defense, October 2011
Olya Ohrimenko	Brown Univ.	PhD Thesis proposal, October 2011
Yanbin Lu	UC-Irvine	PhD Thesis defense, November 2011
Chun Meng	UC-Irvine	Advancement to candidacy, December 2011
Abinesh Ramakrishnan	UC-Irvine	Advancement to candidacy, March 2012
Pegah Sattari	UC-Irvine	PhD Thesis defense, April 2012
Michael Bannister	UC-Irvine	PhD Thesis defense, May 2015
Yingyi Bu	UC-Irvine	PhD Thesis defense, August 2015
Jenny Lam	UC-Irvine	PhD Thesis defense, November 2015
Timothy Johnson	UC-Irvine	Advancement to candidacy, chair, June 2016
Jiayu Xu	UC-Irvine	Advancement to candidacy, November 2016
Sky Faber	UC-Irvine	PhD Thesis defense, November 2016
Juan Jose Besa Vial	UC-Irvine	Advancement to candidacy, chair, March 2017
Ingo van Duijn	Aarhus Univ.	PhD Thesis defense, September 2017
Boyang Wei	UC-Irvine	PhD Thesis defense, August 2018
Pedro Matias	UC-Irvine	Advancement to candidacy, chair, May 2019
Sameera Chayyur	UC-Irvine	Advancement to candidacy, September 2019
Yihan Sun	CMU	PhD Thesis defense, October 2019

University Service:

Ph.D. Requirements Committee, Dept. of Computer Science, chair: 1987–89
Graduate Admissions Committee, Dept. of Computer Science, 1991–1993 (chair: 1992)
Faculty Recruiting Committee, Dept. of Computer Science, 1993,95,96 (chair: 1996)
Steering Committee, Whiting School of Engineering, 1990–93 (chair, 1993)
Johns Hopkins Homewood Academic Computing Oversight Committee, 1990–93
Curriculum Committee, Whiting School of Engineering, 1994–96
Strategic Planning Committee, Whiting School of Engineering, 1999–00
Graduate Policy Committee, UCI Dept. of Information & Computer Science (ICS), 2001–02
Faculty Search Committee in Cryptography, UCI Dept. of ICS, 2001–03
School of Info. and Computer Science Executive Committee, 2002–04
UCI Committee on Educational Policy (CEP), 2002–03, 2004–06
UCI Change of Major Criteria Committee, 2002–03
UCI CEP Policy Subcommittee, 2002–2003
Distinguished Faculty Search Committee, Bren School of ICS, 2004–11 (chair, 2007–08)
Equity Advisor, Bren School of ICS, 2005–09
Dean's Advisory Council, Bren School of ICS, 2007–13
Associate Dean for Faculty Development, Bren School of ICS, 2006–12
Chair, Department of Computer Science, Bren School of ICS, 2012–13
Master of Computer Science Development Committee, Bren School of ICS, 2013–2016

Stragic Planning Committee, Dept. of Computer Science, Bren School of ICS, 2015–16
Master of Computer Science Steering Committee, Bren School of ICS, 2016–
Executive Committee, Bren School of ICS, 2017–18
UC-Irvine Committee on Scholarly Honors & Awards, 2017–
UC-Irvine Special Research Program Review Committee for CalIT2, 2018–19

Courses Taught and Developed:

Advanced Parallel Computing (developed and taught at Hopkins)
Cyber-Puzzlers (designed and taught at UCI)
Computer Literacy (taught at Purdue, developed at Hopkins)
Computer Programming for Scientists and Engineers (taught at Purdue)
Computer Security Algorithms (developed and taught at UCI)
Computational Models (revised and taught at Hopkins)
Computational Geometry (revised and taught at Hopkins and UCI)
Compiler Theory and Design (revised and taught at Hopkins)
Computer Graphics (taught at Hopkins)
Cyber-Fraud Detection and Prevention (designed and taught at UCI)
Data Structures (revised and taught at Hopkins and UCI)
Graph Algorithms (revised and taught at UCI)
Formal Languages and Automata Theory (revised and taught at UCI)
Fundamentals of Algorithms with Applications (revised and taught at UCI)
Introduction to Algorithms (developed and taught at Hopkins and UCI)
Internet Algorithmics (developed and taught at Hopkins, Brown, and UCI)
Design and Analysis of Algorithms (revised and taught at Hopkins and UCI)
Parallel Algorithms (developed and taught at Hopkins and Univ. of Illinois)
Project in Algorithms and Data Structures (revised and taught at UCI)

Scientific Consulting:

APAC Security, Inc., 2005
Algomagic Technologies, Inc., 2000–2005
Army Research Laboratory, Fort Belvior, 1995
AT&T, 1998
Battelle Research Triangle, Columbus Division, 1996
Brown University, 2000–2007
3M, 2015
Purdue University, 2002
The National Science Foundation, 1990–2016
Univ. of Miami, 1999
Walt Disney Animation Studios, 2009

Technical Expert Consulting (last five years):

- 2014–15, Technical expert, deponent, and testifying witness, McKool Smith, PC (Dallas) on behalf of ContentGuard Holdings, in patent litigation.
- 2014–17, Technical expert, Kirkland & Ellis, LLP (Chicago) on behalf of IBM, in patent litigation.
- 2014–16, Technical expert and deponent, Fenwick & West, LLP (San Francisco) on behalf of Symantec, in patent litigation.
- 2016, Technical expert, deponent, and testifying witness, Dentons US LLP., in confidential arbitration.

- 2016, Technical expert, Sheppard Mullin Richter & Hampton LLP (Los Angeles), in non-patent intellectual property dispute.
- 2016–, Technical expert and deponent, Kramer Levin Naftalis & Frankel LLP (Menlo Park, CA) on behalf of Acceleration Bay LLC, in patent litigation.
- 2016–, Technical expert and deponent, Kramer Levin Naftalis & Frankel LLP (Menlo Park, CA) on behalf of Finjan, Inc., in patent litigation.
- 2017–, Technical expert and deponent, Fitzpatrick, Cella, Harper & Scinto, Venable, LLP, on behalf of Koninklijke Philips N.V., in patent litigation.
- 2017–18, Technical expert and deponent, Haynes and Boone, LLP, on behalf of mSIGNIA, in patent litigation.
- 2017–19, Technical expert and deponent, Thompson & Knight LLP on behalf of SEVEN Networks LLC, in patent litigation.
- 2018–, Technical expert, deponent, and testifying witness, Kramer Levin Naftalis & Frankel LLP (Menlo Park, CA) on behalf of Centripetal Networks, in patent litigation.
- 2019–, Technical expert, Reichman Jorgensen LLP on behalf of Kove IO, Inc., in patent litigation.
- 2019–20, Technical expert, McKool Smith on behalf of Excalibur IP, LLC, in patent litigation.
- 2019–, Technical expert, McKool Smith on behalf of SEVEN Networks LLC, in patent litigation.
- 2019–, Technical expert, Kramer Levin Naftalis & Frankel LLP (Menlo Park, CA) on behalf of CUPP Cybersecurity LLC, et al., in patent litigation.
- 2020–, Technical expert, Dovel & Luner LLP on behalf of SimpleAir, Inc., in patent litigation.
- 2020–, Technical expert, Goodwin Procter LLP on behalf of MobileIron, Inc., in patent litigation.

GRANTS AND CONTRACTS

1. PI, "Research Initiation Award: Parallel and Sequential Computational Geometry," National Science Foundation (NSF Grant CCR-8810568), $32,914, 1988–90.
2. co-PI, "Paradigms for Parallel Algorithm Design," NSF and DARPA (as NSF Grant CCR-8908092), $523,837, 1989–93 (with S.R. Kosaraju (PI), S. Kasif, and G. Sullivan).
3. PI, "Parallel Computation and Computational Geometry," NSF (Grant CCR-9003299), $67,436, 1990–93.
4. co-PI, "A Facility for Experimental Validation," NSF (Grant CDA-9015667), $1,476,147, 1991–96 (with G. Masson (PI), J. Johnstone, S. Kasif, S.R. Kosaraju, S. Salzberg, S. Smith, G. Sullivan, L. Wolff, and A. Zwarico).
5. PI, "Parallel Network Algorithms for Cell Suppression," The Bureau of the Census (JSA 91-23), $14,998 1991–92.
6. PI, "A Geometric Framework for the Exploration & Analysis of Astrophysical Data," NSF (Grant IRI-9116843), $535,553, 1991–96 (with S. Salzberg and H. Ford (from Physics and Astronomy Dept.)).
7. PI, "Research Experiences for Undergraduates supplement to IRI-9116843," NSF, $4,000, 1993–94 (with S. Salzberg and H. Ford).
8. PI, "Constructing, Maintaining, and Searching Geometric Structures," NSF (Grant CCR-9300079), $134,976, 1993–96.
9. co-PI, "Robust and Applicable Geometric Computing," Army Research Office (ARO MURI Grant DAAH04-96-1-0013), $4,500,000, 1996–2000 (with F. Preparata (PI, Brown U.), R. Tamassia (Brown U.), S. Rao Kosaraju, J. Vitter (Duke U.), and P. Agarwal (Duke U.)).

Subaward size: $1,466,640.

10. PI, "Application-Motivated Geometric Algorithm Design," NSF (Grant CCR-9625289), $107,389, 1996-98.
11. co-PI, "vBNS Connectivity for the Johns Hopkins University," NSF, $350,000, 1997–99 (with T.O. Poehler (PI), D.J. Binko, J.G. Neal, and A.S. Szalay).
12. co-PI, "Product Donation, Technology for Education Program," Intel Corporation, $480,071, 1997–2001 (with T.O. Poehler (PI), J.H. Anderson, A.S. Szalay, and M. Robbins).
13. co-PI, "A Networked Computing Environment for the Manipulation & Visualization of Geometric Data" (Research Infrastructure), NSF, $1,638,785, 1997–2003 (with L.B. Wolff (PI), Y. Amir, S.R. Kosaraju, S. Kumar, R. Tamassia (Brown U.), R.H. Taylor, and D. Yarowsky).
14. PI, "Geometric Algorithm Design and Implementation," NSF, Grant CCR-9732300, $224,982, 1998–2002.
15. PI, "Certification Management Infrastructure – Certificate Revocation," $52,023, 1998, NSA LUCITE grant.
16. PI, "Software Engineering Data Loading, Analysis, and Reporting," $41,614, 1998, NSA LUCITE grant.
17. PI, "Establishing a LUCITE Collaboration Environment," $10,018, 1998, NSA LUCITE grant.
18. PI, "In Support of a Secure Multilingual Collabortive Computing Environment," $51,471, 1999-2000, NSA LUCITE grant.
19. PI, "Accessing Large Distributed Archives in Astronomy and Particle Physics," $199,981. subcontract to UCI from Johns Hopkins Univ. on NSF Grant PHY-9980044 (total budget, $2,500,000), 1999–2004.
20. PI, "Efficient and Scalable Infrastructure Support for Dynamic Coalitions," $1,495,000, DARPA Grant F30602-00-2-0509, 2000-2003 (with Robert Cohen and Roberto Tamassia), including $227,893 subaward to UCI (with Gene Tsudik).
21. PI, "Graph Visualization and Geometric Algorithm Design," $400,000, NSF Grant CCR-0098068, 2001-2004 (with Roberto Tamssia).
22. PI, "Collaborative Research: Teaching Data Structures to the Millennium Generation," $125,00, NSF Grant DUE-0231467, 2003–2005.
23. PI, "Collaborative Research: An Algorithmic Approach to Cyber-Security," $100,000, NSF Grant CCR-0311720, 2003–2006.
24. PI, "The OptIPuter," $900,000, subcontract from UCSD on NSF ITR grant CCR-0225642 (total budget, $13.5 million), 2002–2007 (with Padhraic Smyth and Kane Kim).
25. PI, "ITR: Algorithms for the Technology of Trust," $300,000, NSF Grant CCR-0312760, 2003–2009.
26. co-PI, "SDCI Data New: Trust Management for Open Collaborative Information Repositories: The CalSWIM Cyberinfrastructure," NSF grant OCI-0724806, $1,103,590, 2007–2012.
27. co-PI, "Support for Machine Learning Techniques for Cyber-Fraud Detection," Experian Corporation, $200,000 gift, 2008.
28. PI, "IPS: Collaborative Research: Privacy Management, Measurement, and Visualization in Distributed Environments," NSF Grant IIS-0713046, $224,851, 2007–2009.
29. PI, "Collaborative Research: Algorithms for Graphs on Surfaces," $400,000, NSF Grant CCR-0830403, 2008–2011.
30. PI, "ROA Supplement: IPS: Collaborative Research: Privacy Management, Measurement, and Visualization in Distributed Environments," NSF Grant IIS-0847968, $25,000, 2008–2009.

31. co-investigator, "Scalable Methods for the Analysis of Network-Based Data," Office of Naval Research: Multidisciplinary University Research Initiative (MURI) Award, number N00014-08-1-1015, $529,152, 2008–2014.
32. PI, "EAGER: Usable Location Privacy for Mobile Devices," NSF Grant 0953071, $300,000, 2009–2011.
33. PI, "TC:Large:Collaborative Research: Towards Trustworthy Interactions in the Cloud," NSF Grant 1011840, $500,000, 2010-2015.
34. PI, "TWC: Medium: Collaborative: Privacy-Preserving Distributed Storage and Computation," NSF Grant 1228639, $390,738, 2012-2018.
35. PI, "Support for Research on Geometric Motion Planning," 3M Corporation, $40,000 gift, 2014.
36. PI, "A4V: Automated Analysis of Algorithm Attack Vulnerabilities," subcontract 10036982-UCI from University of Utah for DARPA agreement no. AFRL FA8750-15-2-0092, $980,000, 2015–2019.
37. PI, "TWC: Small: Collaborative: Practical Security Protocols via Advanced Data Structures," NSF Grant 1526631, $166,638, 2015–2018.
38. PI, "NSF-BSF: AF: Small: Geometric Realizations and Evolving Data," NSF Grant 1815073, $474,392, 2018–2021.

SELECTED INVITED TALKS (RECENT YEARS ONLY)

- "Probabilistic Packet Marking for Large-Scale IP Traceback," Purdue Univ., 2003
- "Algorithms for Data Authentication," Harvey Mudd College, 2003
- "Efficient Tree-Based Revocation in Groups of Low-State Devices," Univ. of Arizona, 2004
- "Leap-Frog Packet Linking and Diverse Key Distributions for Improved Integrity in Network Broadcasts," Southern California Security and Cryptography Workshop, 2005
- "Is Your Business Privacy Protected?," NEXT Connections, 2005
- "Distributed Peer-to-peer Data Structures," Harvard Univ., 2006
- "Balancing Life with an Academic Research Career," Grace Hopper Conference, 2006
- "Computer Security in the Large," Univ. Texas, San Antonio, 2006
- "Inspirations in Parallelism and Computational Geometry," Brown Univ., 2006
- "Efficiency and Security Issues for Distributed Data Structures," Computer Science Distinguished Lecture Series, Johns Hopkins Univ., 2006
- "Efficiency and Security Issues for Distributed Data Structures," UCLA, 2006
- "Efficiency and Security Issues for Distributed Data Structures," Edison Distinguished Lecturer Series, Univ. of Notre Dame, 2006
- "Efficiency and Security Issues for Distributed Data Structures," Computer Science Distinguished Lecturer Series, Texas A & M Univ., 2006
- "Algorithms for Secure Computing and Searching with Applications to Medical Informatics," Purdue Univ., 2006
- "Blood on the Computer: How Algorithms for Testing Blood Samples can be Used for DNA Sequencing, Wireless Broadcasting, and Network Security," Univ. of Southern California, 2007
- "Blood on the Computer: How Algorithms for Testing Blood Samples can be Used for DNA Sequencing, Wireless Broadcasting, and Network Security," Univ. California, San Diego, 2007
- "Blood on the Computer: How Algorithms for Testing Blood Samples can be Used for DNA Sequencing, Wireless Broadcasting, and Network Security," Univ. Minnesota, 2007

- "Blood on the Database: How Algorithms for Testing Blood Samples can be Used for Database Integrity," Invited Keynote, 21st Annual IFIP WG 11.3 Working Conference on Data and Applications Security (DBSec), 2007
- "Space-Efficient Straggler Identification," ALCOM Seminar, Univ. of Aarhus, 2007
- "Blood on the Computer: How Algorithms for Testing Blood Samples can be used in Modern Applications," ALCOM Seminar, Univ. of Aarhus, 2007
- "Studying Road Networks Through an Algorithmic Lens," ALCOM Seminar, Univ. of Aarhus, 2008
- "Studying Geometric Graph Properties of Road Networks Through an Algorithmic Lens," International Workshop on Computing: from Theory to Practice, 2009
- "Randomized Shellsort: A Simple Oblivious Sorting Algorithm," Distinguished Lecture Series, Department of Computer Science, Brown University, 2009
- "Simulating Parallel Algorithms in the MapReduce Framework with Applications to Parallel Computational Geometry," MASSIVE 2010
- "Data Cloning Attacks for Nearest-Neighbor Searching based on Retroactive Data Structures," Department of Computer Science, UCSB, 2011
- "Turning Privacy Leaks into Floods: Surreptitious Discovery of Social Network Friendships and Other Sensitve Binary Attribute Vectors," Department of Computer Science Distinguished Lecturer Series, Univ. of Illinois, Chicago, 2011
- "Turning Privacy Leaks into Floods: Surreptitious Discovery of Social Network Friendships and Other Sensitve Binary Attribute Vectors," Department of Computer Science, Purdue Univ., 2011
- "Spin-the-bottle Sort and Annealing Sort: Oblivious Sorting via Round-robin Random Comparisons," Department of Computer Science, Brown Univ., 2012
- "Using Data-Oblivious Algorithms for Private Cloud Storage Access," Qatar University, 2013
- "Using Data-Oblivious Algorithms for Private Cloud Storage Access," Department of Computer Science and Engineering Distinguished Speaker Series, University of Buffalo, 2013
- "Force-Directed Graph Drawing Using Social Gravity and Scaling," invited talk, ICERM Workshop on Stochastic Graph Models, Providence, RI, 2014
- "Invertible Bloom Lookup Tables and Their Applications in Large-Scale Data Analysis," invited key-note speaker, Algorithms for Big Data, Frankfurt, Germany, 2014
- "Invertible Bloom Lookup Tables and Their Applications in Large-Scale Data Analysis," Brown University, Providence, RI, 2014
- "Studying Road Networks Through an Algorithmic Lens," Bold Aspirations Visitor and Lecture, University of Kansas, 2015
- "Learning Character Strings via Mastermind Queries, with Case Studies," Invited Lecture, Workshop on Pattern Matching, Data Structures and Compression, Bar-Ilan University, Tel Aviv, Israel, 2016
- "Invertible Bloom Lookup Tables and Their Applications in Data Analysis," University of Hawaii, 2016
- "Invertible Bloom Lookup Tables," Purdue University, 2016
- "Combinatorial Pair Testing: Distinguishing Workers from Slackers," Calvin Univ., 2016
- "Invertible Bloom Lookup Tables," University of California, Riverside, 2016
- "2-3 Cuckoo Filters for Faster Triangle Listing and Set Intersection," Technion, Israel Institute of Technology, Haifa, Israel, 2017
- "2-3 Cuckoo Filters for Faster Triangle Listing and Set Intersection," University of Arizona,

2017

- "Parallel Computational Geometry," First Hawaii Workshop on Parallel Algorithms and Data Structures, University of Hawaii, 2017
- "Fighting Gerrymandering with Algorithmic Fairness," Calvin University, 2019
- "Fighting Gerrymandering with Algorithmic Fairness," Carnegie Mellon University, 2019
- "Sorting Evolving Data in Parallel," Second Hawaii Workshop on Parallel Algorithms and Data Structures, University of Hawaii, 2019

PROFESSIONAL SOCIETY MEMBERSHIPS

American Association for the Advancement of Science (AAAS), Fellow
IEEE and IEEE Computer Society, Fellow
Association for Computing Machinery (ACM), Fellow

TECHNICAL EXPERT CONSULTING SUMMARY
(LAST FIVE YEARS)

Michael T. Goodrich, PhD

Dept. of Computer Science
Bren School of Info. & Computer Sciences
University of California, Irvine
Irvine, CA 92697-3435

Phone: (949)824-9366
http://www.ics.uci.edu/~goodrich/
E-mail: goodrich (at) acm.org

TECHNICAL EXPERT CONSULTING

The following is a listing of prior and current expert consulting work for Michael T. Goodrich, PhD, for the last five years, but it may not include confidential information, as per written agreements.

1. Mar. 2014 to Nov. 2015, Technical expert, deponent, and testifying expert at two trials, retained by McKool Smith, PC (Dallas) on behalf of ContentGuard Holdings, in patent litigation, ContentGuard Holdings v. Amazon, Apple, HTC, Samsung, Huawei, BlackBerry, and Motorola (Google). Civil Actions No. 2:13-cv-01112-JRG and 2:14-cv-00061-JRG (E.D. Tex.).

2. Apr. 2014 to Aug. 2017, Technical expert and deponent, retained by Kirkland & Ellis, LLP (Chicago), on behalf of IBM, through IMS Expert Services, in patent litigation, PersonalWeb and Level 3 Communications, LLC v. IBM. Civil Action No. 6:12-cv-00661 (E.D. Tex.).

3. Oct. 2014 to Feb. 2016, Technical expert and deponent, retained by Fenwick & West, LLP (San Francisco) on behalf of Symantec, through DOAR Expert Services, in several IPRs for patent litigation, Columbia University v. Symantec. IPR2015-00370, IPR2015-00371, and IPR2015-00372 (PTAB).

4. Feb. 2016 to May 2016, Technical expert, deponent, and testifying witness in non-patent arbitration hearing, retained by Dentons US LLP in confidential arbitration.

5. Apr. 2016 to present, Technical expert and deponent, retained by Kramer Levin LLP on behalf of Acceleration Bay LLC, in IPR proceedings and patent litigation, Acceleration Bay LLC v. Activision Bizzard, Inc., Electronic Arts Inc., Take-Two Interactive Software, Inc., Rockstar Games, Inc., and 2K Sports, Inc. Civil Actions No. 1:16-cv-00453-RGA, No. 1:16-cv-00454-RGA, No. 1:16-cv-00455-RGA (D. Del.), IPR2015-01951, IPR2015-01953, IPR2015-01964, IPR2015-01970, IPR2015-01972, IPR2015-01996, IPR2016-00724, and IPR2016-00747 (PTAB).

6. May 2016 to present, Technical expert and deponent, retained by Kramer Levin LLP on behalf of Finjan, Inc., in PTO proceedings and patent litigations, Finjan, Inc. v. Blue Coat,

Finjan, Inc. v. ESET, LLC, Finjan, Inc. v. Juniper Networks, Inc., Finjan, Inc. v. Palo Alto Networks, Inc., Finjan, Inc. v. Cisco Systems, Inc., Finjan, Inc. v. Qualys, Inc., Finjan, Inc. v. Rapid7, Inc., and Finjan, Inc. v. SonicWall, Inc., IPR2015-01974, IPR2015-01979, IPR2016-00478, IPR2016-00159 (PTAB), and Civil Actions No. 5:13-cv-03999-BLF (N.D. Cal.), 5:15-cv-03295-BLF-PSG (N.D. Cal.), 3:17-cv-05659-WHA (N.D. Cal.), 17-cv-0072-BLF-SVK (N.D. Cal.), 4:18-cv-07229-YGR (N.D. Cal.), 18-1519-MN (Del.), and 5:17-cv-04467-BLF (N.D. Cal.).

7. Jun. 2016 to Oct. 2016, Technical expert, retained by Sheppard Mullin Richter & Hampton LLP in confidential non-patent intellectual property dispute.

8. Jan. 2017 to present, Technical expert and deponent, retained by Fitzpatrick, Cella, Harper & Scinto, Venable LLP, on behalf of Koninklijke Philips N.V. and U.S. Philips Corp., in patent litigation, Philips v. Acer Inc., ASUSTeK Computer, Double Power Tech., HTC, Southern Telecom, Visual Land, Zowee Marketing Co., Ltd., Shenzen Zowee Technology Co., Ltd., YiFang USA, Inc. d/b/a EFun, Inc., and Intervenor/Counterclaim Defendant Microsoft Corporation. Civil Actions No. 15-1125-GMS, No. 15-1126-GMS, No. 15-1127-GMS, No. 15-1128-GMS, No. 15-1130-GMS, No. 15-1131-GMS, No. 15-1170-GMS (D. Del.), and 4:18-cv-01885-HSG (N.D. Cal., Oakland Div.).

9. Jul. 2017 to Dec. 2018, Technical expert and deponent, retained by Haynes and Boone, LLP, on behalf of mSIGNIA, in patent litigation, mSIGNIA v. InAuth. Civil Action No. 8:17-cv-01289-AG-KES (C.D. Cal.).

10. Nov. 2017 to Jun. 2019, Technical expert and deponent, retained by Thompson & Knight LLP on behalf of SEVEN Networks LLC, in patent litigation, SEVEN Networks LLC v. Google, Samsung, and ZTE. Civil Actions No. 2:17-CV-441-JRG, No. 2:17-CV-442-JRG (E.D. Texas), and 3:17-cv-1495-M (N.D. Texas).

11. Jun. 2018 to present, Technical expert, deponent, and testifying witness, retained by Kramer Levin LLP on behalf of Centripetal Networks, Inc., in IPRs and patent litigation, Centripetal Networks, Inc. v. Keysight Technologies, Inc. and Ixia, and Cisco. Civil Actions No. 2:17-cv-00383/HCM-LRL and 2:18-cv-00094-HCM-LRL (E.D. Virginia, Norfolk Div.), IPR2018-01386, IPR2018-01443, IPR2018-01444, IPR2018-01512, and IPR2018-01513 (PTAB).

12. Feb. 2019 to present, Technical expert, retained by Reichman Jorgensen LLP on behalf of Kove IO, Inc., in patent litigation, Kove IO, Inc. v. Amazon Web Services, Inc., Civil Action No. 1:18-cv-08175 (N. Dist. Illinois, Eastern Div.).

13. Apr. 2019 to present, Technical expert, retained by McKool Smith on behalf of SEVEN Networks LLC in patent litigation, SEVEN Networks LLC v. Apple, Inc. Civil Action No. 2:19-cv-115 (E.D. Texas).

14. Jun. 2019 to present, Technical expert, retained by Kramer Levin LLP on behalf of CUPP Cybersecurity LLC, in patent litigation, CUPP Cybersecurity LLC, et al., v. Trend Micro, Inc., et al. Civil Action 3:18-cv-01251-M (N.D. Tex.)

15. Apr. 2020 to present, Technical expert, retained by Dovel & Luner LLP on behalf of SimpleAir, Inc., in patent litigation, SimpleAir, Inc. v. Google LLC. Civil actions 2:20-cv-2839-JAK-PLA and 2:16-cv-3758-JAK-PLA (C.D. Cal.).

16. Jul. 2020 to present, Technical expert, retained by Goodwin Procter LLP on behalf of MobileIron, Inc., in patent litigation, MobileIron, Inc. v. BlackBerry Corp. Civil action no. 3:20-cv-02877-JCS (N.D. Cal.).