

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re Patent of: Vipul Pabari Attorney Docket No. 42342-0156IP1
U.S. Patent No.: 11,656,915
Issue Date: May 23, 2023
Appl. Serial No.: 17/074,500
Filing Date: October 19, 2020
Title: VIRTUAL SYSTEMS MANAGEMENT

DECLARATION OF MICHAEL MITZENMACHER

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

Date: 3/6/2026

By: 
Michael Mitzenmacher

Table of Contents

I.	ASSIGNMENT.....	11
II.	QUALIFICATIONS AND BACKGROUND INFORMATION.....	11
III.	OVERVIEW OF CONCLUSIONS FORMED	17
IV.	LEVEL OF ORDINARY SKILL IN THE ART.....	18
V.	LEGAL STANDARDS	19
	A. Terminology.....	19
	B. Legal Standards.....	20
	C. Anticipation.....	20
	D. Obviousness	21
VI.	MATERIALS CONSIDERED	25
VII.	THE '915 PATENT.....	25
	A. Overview of the '915 Patent	25
	B. Prosecution History of the '915 Patent.....	27
VIII.	OVERVIEW OF THE PRIOR ART	41
	A. Khandekar (MS-1004)	43
	B. Hirschfeld (MS-1006).....	47
	C. Bandhole (MS-1007)	53
	D. Vasilevsky (MS-1005).....	56
	E. Fujino (MS-1008)	59
	F. Smith (MS-1010).....	62
IX.	OVERVIEW OF THE PRIOR ART COMBINATIONS	65
	A. The Combination Of Khandekar And Bandhole	65
	B. The Combination Of Khandekar And Vasilevsky.....	67
	C. The Combination Of Khandekar And Fujino	72
	D. The Combination of Hirschfeld And Smith	75
	E. The Combination of Hirschfeld, Smith, And Vasilevsky.....	79
	F. The Combination of Hirschfeld, Smith, And Fujino	82
X.	THE PRIOR ART MAKES THE CHALLENGED CLAIMS OBVIOUS...90	
	A. Grounds 1A/1B: Claims 1 And 2 Are Anticipated And/Or Rendered Obvious By Khandekar.....	90
	1. Claim 1	90
	2. Claim 2	102
	B. Ground 1C: Claims 3-5 Are Obvious Over The Combination of Khandekar And Bandhole.....	102
	1. Claim 3	102

2.	Claim 4	103
3.	Claim 5	105
C.	Ground 1D: Claims 6-8 Are Obvious Over The Combination of Khandekar And Vasilevsky	107
1.	Claim 6	107
2.	Claim 7	108
3.	Claim 8	109
4.	Claim 11	109
5.	Claim 12	114
6.	Claim 13	115
7.	Claim 14	115
D.	Ground 1E: Claims 9 And 10 Are Obvious Over The Combination of Khandekar And Fujino.....	117
1.	Claim 9	117
2.	Claim 10	118
E.	Ground 2A: Claims 1 And 2 Are Obvious Over The Combination of Hirschfeld And Smith	119
1.	Claim 1	119
2.	Claim 2	132
F.	Ground 2C: Claims 6-8 And 11-14 Are Obvious Over The Combination of Hirschfeld, Smith, And Vasilevsky	133
1.	Claim 6	140
2.	Claim 7	141
3.	Claim 8	142
4.	Claim 11	143
5.	Claim 12	155
6.	Claim 13	157
7.	Claim 14	159
G.	Ground 2D: Claims 9 And 10 Are Obvious Over The Combination of Hirschfeld, Smith, And Fujuno.....	161
1.	Claim 9	161
2.	Claim 10	164
XI.	CONCLUSION.....	166

LISTING OF CLAIMS

Claim 1	
[1.pre]	A method for access to one or more virtualization system resources over a network, the method comprising:
[1.1]	implementing a virtualization system that has a control computer and one or more physical resources and virtual assets connected to the control computer,
[1.2]	wherein each of the one or more physical resources execute at least two virtual assets that comprise software instantiations of computer systems, and
[1.3]	the virtual assets are managed by virtualization layer instances running on each of the physical resources;
[1.4]	connecting an end user to the control computer; and
[1.5]	accessing, through the control computer connection, the virtual assets by the end user.
Claim 2	
[2]	The method of claim 1, wherein connecting the end user to the control computer further comprises connecting the end user to the control computer over one of a network, a wide area network or the Internet.
Claim 3	

[3]	The method of claim 2, wherein connecting the end user to the control computer further comprises providing the end user, a desktop, a laptop or a personal digital assistant (PDA) and client resources.
Claim 4	
[4]	The method of claim 3, wherein accessing the virtual assets further comprising accessing an application server by the end user.
Claim 5	
[5]	The method of claim 1, wherein the virtual asset is providing end user access to one of an application server, to a middleware server and to a database.
Claim 6	
[6]	The method of claim 1, wherein the virtual asset is providing a virtual storage resource.
Claim 7	
[7]	The method of claim 6, wherein the virtual storage resource further comprises mapping a logical storage unit to a physical storage disk.
Claim 8	

[8]	The method of claim 7, wherein providing the virtual storage resource further comprises mapping a logical storage unit to a physical storage disk.
Claim 9	
[9]	The method of claim 1, wherein providing the virtual assets further comprise providing a logical network unit to a physical storage disk.
Claim 10	
[10]	The method of claim 9, wherein providing the logical network resource further comprises mapping a logical network unit to a physical network device.
Claim 11	
[11.pre]	A method, comprising:
[11.1]	communicating, by a user, over a network to access one or more virtual assets running in a virtualization system,
[11.2]	wherein the virtualization system includes a control computer and one or more physical resources,
[11.3]	wherein each of the one or more physical resources execute at least two virtual assets that comprise software instantiations of computer systems and

[11.4]	the virtual assets are managed by virtualization layer instances running on each of the one or more physical resources;
[11.5]	requesting, by the user, provisioning of a virtual asset of the one or more virtual assets in the virtualization system; and
[11.6]	wherein the provisioned virtual asset is a virtual storage and the virtual storage comprises one or more logical storage units mapped to one or more physical storage drives.
Claim 12	
[12]	The method of claim 11 further comprising requesting, by a management application programming interface (API) of the virtualization system, the provisioning of the virtual storage.
Claim 13	
[13]	The method of claim 12 further comprising configuring, by the user, the virtual storage.
Claim 14	
[14]	The method of claim 13 further comprising receiving, by the user, an inventory of the virtual storage.
Claim 1	
[1.pre]	A method for access to one or more virtualization system resources over a network, the method comprising:

[1.1]	implementing a virtualization system that has a control computer and one or more physical resources and virtual assets connected to the control computer,
[1.2]	wherein each of the one or more physical resources execute at least two virtual assets that comprise software instantiations of computer systems, and
[1.3]	the virtual assets are managed by virtualization layer instances running on each of the physical resources;
[1.4]	connecting an end user to the control computer; and
[1.5]	accessing, through the control computer connection, the virtual assets by the end user.
Claim 2	
[2]	The method of claim 1, wherein connecting the end user to the control computer further comprises connecting the end user to the control computer over one of a network, a wide area network or the Internet.
Claim 3	
[3]	The method of claim 2, wherein connecting the end user to the control computer further comprises providing the end user, a desktop, a laptop or a personal digital assistant (PDA) and client resources.
Claim 4	
[4]	The method of claim 3, wherein accessing the virtual assets further comprising accessing an application server by the end user.
Claim 5	
[5]	The method of claim 1, wherein the virtual asset is providing end user access to one of an application server, to a middleware server and to a database.
Claim 6	

[6]	The method of claim 1, wherein the virtual asset is providing a virtual storage resource.
Claim 7	
[7]	The method of claim 6, wherein the virtual storage resource further comprises mapping a logical storage unit to a physical storage disk.
Claim 8	
[8]	The method of claim 7, wherein providing the virtual storage resource further comprises mapping a logical storage unit to a physical storage disk.
Claim 9	
[9]	The method of claim 1, wherein providing the virtual assets further comprise providing a logical network unit to a physical storage disk.
Claim 10	
[10]	The method of claim 9, wherein providing the logical network resource further comprises mapping a logical network unit to a physical network device.
Claim 11	
[11.pre]	A method, comprising:
[11.1]	communicating, by a user, over a network to access one or more virtual assets running in a virtualization system,
[11.2]	wherein the virtualization system includes a control computer and one or more physical resources,
[11.3]	wherein each of the one or more physical resources execute at least two virtual assets that comprise software instantiations of computer systems and
[11.4]	the virtual assets are managed by virtualization layer instances running on each of the one or more physical resources;

[11.5]	requesting, by the user, provisioning of a virtual asset of the one or more virtual assets in the virtualization system; and
[11.6]	wherein the provisioned virtual asset is a virtual storage and the virtual storage comprises one or more logical storage units mapped to one or more physical storage drives.
Claim 12	
[12]	The method of claim 11 further comprising requesting, by a management application programming interface (API) of the virtualization system, the provisioning of the virtual storage.
Claim 13	
[13]	The method of claim 12 further comprising configuring, by the user, the virtual storage.
Claim 14	
[14]	The method of claim 13 further comprising receiving, by the user, an inventory of the virtual storage.

I. ASSIGNMENT

1. I have been retained on behalf of Microsoft Corporation (“Microsoft”) to offer technical opinions related to 11,656,915 (“the ‘915 patent”). MS-1001. I understand that Microsoft is requesting that the Patent Trial and Appeal Board (“PTAB” or “Board”) to institute an *inter partes* review (“IPR”) proceeding of the ‘915 patent.

2. I have been asked to provide my independent analysis of the ‘915 patent based on the prior art publications cited in this declaration.

3. I received no compensation for this declaration beyond my normal hourly compensation based on my time actually spent analyzing the ‘915 patent, the prior art publications cited below, and issues related thereto, and I will not receive any added compensation based on the outcome of any IPR or other proceeding involving the ‘915 patent.

II. QUALIFICATIONS AND BACKGROUND INFORMATION

4. I am over the age of 18 and am competent to write this declaration. I have personal knowledge, or have developed knowledge of these technologies based upon education, training, or experience, of the matters set forth herein. My qualifications are summarized here and explained in more detail in my curriculum vitae, which is attached as Appendix A to this report. Appendix A also includes a list of my publications and the cases in which I have testified at deposition,

hearing, or trial within the past four years.

5. I am currently employed as a Professor of Computer Science at Harvard University. Specifically, I am the Thomas J. Watson, Sr. Professor of Computer Science in the School of Engineering and Applied Sciences. I joined the faculty at Harvard as an Assistant Professor in January 1999. I was promoted to Associate Professor in 2002 and to Professor in 2005. In 2010, I began a three-year term as Area Dean—equivalent to what other schools call Department Chair—of Computer Science and held that position through June 2013. I also served as Area Co-Chair of Computer Science for the 2018-2019 academic year. From August 1996 to January 1999, I was employed as a Research Scientist at Digital Systems Research Center in Palo Alto, where my work included projects on algorithms for the Internet, including information retrieval on the Web, erasure codes, error-correcting codes, on-line algorithms, and load balancing.

6. I teach courses on various computer science topics to undergraduate and graduate students at Harvard. I have regularly taught the undergraduate class on algorithms and data structures. I have also regularly taught graduate level classes involving algorithms and data structures, that include topics related to machine learning. For example, in both courses I have covered the area of learning-augmented algorithms, where algorithms use predictions to improve their performance.

7. I earned my undergraduate degree in Mathematics and Computer Science from Harvard College in 1991, a Certificate of Advanced Study in Mathematics from Cambridge University in 1992, and a Ph.D. in Computer Science from the University of California at Berkeley in 1996.

8. I have published over 250 research papers in computer science conferences and journals. I am listed as an inventor or co-inventor on 19 issued patents covering computer technology, and I am the co-author of a textbook entitled “Probability and Computing: Randomized Algorithms and Probabilistic Analysis” published by Cambridge University Press. In addition, I have consulted for several companies as a professor, such as Microsoft, Mitsubishi, Akamai, ITA Software, Adverplex, Fluent Mobile, and e-Harmony.

9. I have received numerous professional recognitions from my peers. I am a Fellow of the Association for Computing Machinery (“ACM”) and a Fellow of the Institute of Electrical and Electronics Engineers (“IEEE”), both of which are the two primary professional organizations for computer scientists. ACM Fellows, for example, are described as follows: “ACM’s most prestigious member grade recognizes the top 1% of ACM members for their outstanding accomplishments in computing and information technology and/or outstanding service to ACM and the larger computing community.” See <https://awards.acm.org/fellows> (“About ACM Fellows”). My citation for my induction as an ACM Fellow is: “For contributions

to coding theory, hashing algorithms and data structures, and networking algorithms.” Among my awards, I shared in receiving the 2020 ACM Paris Kanellakis Theory and Practice Award for the discovery and analysis of balanced allocations, known as the power of two choices, and their extensive applications to practice.

10. As mentioned, I am the co-author of a textbook entitled “Probability and Computing: Randomized Algorithms and Probabilistic Analysis,” which was published by Cambridge University Press. This textbook is used in advanced undergraduate or beginning graduate courses at Brown University, Harvard University, the University of California at Berkeley, the University of Victoria, Tufts University, the University of Massachusetts at Amherst, Purdue University, the University of Pennsylvania, and several other universities. The second edition added additional sections related to topics in machine learning.

11. I have worked for many years on the foundations and applications of Artificial Intelligence, with recent research examining the ways to improve the efficiency, reliability, and security of modern AI computing systems, including using AI predictions to improve algorithms and data structures. My experience spans large language models (LLMs), distributed and federated learning, security issues, and specific applications such as AI systems and models for biological analyses.

12. My curriculum vitae, included as an appendix to this declaration, includes a list of publications on which I am a named author. It contains further details regarding my experience, education, publications, and other qualifications to render an expert opinion in connection with this proceeding.

13. In writing this Declaration, I have considered the following: my own knowledge and experience, including my work experience in the fields of computer network virtualization, automated deployment of virtual machines, and distributed systems administration; my experience in teaching those subjects; and my experience in working with others involved in those fields. In addition, I have analyzed the following publications and materials, in addition to other materials I cite in my declaration:

14. I have reviewed the '915 patent (MICROSOFT-1001), and relevant excerpts of the prosecution history (MICROSOFT-1002) of the '915 patent. Among various textbooks, documents, and publications, I have also reviewed the following prior art references:

- U.S. Pat. No. 7,577,722 ("Khandaker") (MS-1004)
- U.S. Pat. App. No. 2005/0044301 ("Vasilevsky") (MS-1005)
- U.S. Pat. No. 6,880,002 ("Hirschfeld") (MS-1006)
- U.S. Pat. App. No. 2002/0049803 ("Bandhole") (MS-1007)
- U.S. Pat. No. 7,457,925 ("Fujino") (MS-1008)

- Int. Pat. Pub. No. WO 02/03220 (“Tremain”) (MS-1009)
- Smith, James E. & Ravi Nair, “Virtual Machines: Versatile Platforms for Systems and Processes,” Morgan Kaufmann, 2005, ScienceDirect (“Smith”) (MS-1010)

15. Counsel (Fish & Richardson) has informed me that I should consider these materials through the lens of one of ordinary skill in the art related to the ‘915 patent at the time of the earliest possible priority date of the ‘915 patent, and I have done so during my review of these materials. I have been informed by Counsel to use the date of 8/15/2005 as the Priority Date.

16. I have no financial interest in the outcome of this proceeding. I am being compensated for my work as an expert on an hourly basis. My compensation is not dependent on the outcome of these proceedings or the content of my opinions.

17. My opinions, as explained below, are based on my education, experience, and expertise in the fields relating to the ‘915 Patent. Unless otherwise stated, my testimony below refers to the knowledge of one of ordinary skill in the art as of the earliest possible priority date. Any figures that appear within this document have been prepared with the assistance of Counsel and reflect my understanding of the ‘915 Patent and the prior art discussed below.

III. OVERVIEW OF CONCLUSIONS FORMED

18. This declaration explains the conclusions that I have formed based on my analysis of all of the considered materials and my personal experience. To summarize those conclusions, based upon my knowledge and experience and my review of the prior art references listed above, I believe that:

- Ground 1A/1B: Claims 1 and 2 are anticipated and/or rendered obvious over Khandekar
- Ground 1C: Claims 3-5 are obvious over the combination of Khandekar and Bandhole
- Ground 1D: Claims 6-8 and 11-14 are obvious over the combination of Khandekar and Vasilevsky
- Ground 1E: Claims 9 and 10 are obvious over the combination of Khandekar and Fujino
- Ground 2A: Claims 1 and 2 rendered obvious over the combination of Hirschfeld and Smith
- Ground 2B: Claims 3-5 are obvious over the combination of Hirschfeld, Smith, and Bandhole
- Ground 2C: Claims 6-8 and 11-14 are obvious over the combination of Hirschfeld, Smith, and Vasilevsky

- Ground 2D: Claims 9 and 10 are obvious over the combination
of Hirschfeld, Smith, and Fujino

19. In support of these conclusions, I provide an overview of the references in Section IX and more detailed comments regarding the obviousness of claims 1-14 (“the Challenged Claims”) of the ‘915 patent in Section X.

IV. LEVEL OF ORDINARY SKILL IN THE ART

20. I have been informed that a person of ordinary skill in the art (“POSITA”) is a hypothetical person who is presumed to have the skill and experience of an ordinary worker in the field at the time of the alleged invention. Because I do not know at what date the invention as claimed was made, if ever, I have used the Critical Date of the ‘915 patent as the point in time for claim interpretation purposes. My opinion does not change if the invention date is earlier.

21. Based on my knowledge and experience in the field and my review of the ‘915 patent and file history, I believe that a person of ordinary skill in the art in this matter would have had (1) a bachelor’s degree in computer science, computer engineering, electrical engineering, or a related field, and (2) two years of industry experience relating to virtual computing systems. A greater amount of education, i.e., a doctorate in an academic area or a related technical field, would compensate for fewer years of work experience.

22. Based on my experiences, I have a good understanding of the

capabilities of a POSITA. Indeed, I have taught, participated in organizations, and worked closely with many such persons over the course of my career. Based on my knowledge, skill, and experience, I have an understanding of the capabilities of a POSITA. For example, from my industry experience, I am familiar with what a POSITA would have known and found predictable in the art. From teaching and supervising my post-graduate students, I also have an understanding of the knowledge that a person with this academic experience possesses. Furthermore, I possess those capabilities myself.

V. LEGAL STANDARDS

A. Terminology

23. I have been informed by Counsel and understand that the best indicator of claim meaning is its usage in the context of the patent specification as understood by one of ordinary skill. I further understand that the words of the claims should be given their plain meaning unless that meaning is inconsistent with the patent specification or the patent's history of examination before the Patent Office. Counsel has also informed me, and I understand that, the words of the claims should be interpreted as they would have been interpreted by one of ordinary skill at the time of the invention was made (not today). The '915 patent was filed 10/19/2020, and claims priority to 8/15/2005, which I refer to as the "Critical Date" in this proceeding. Because I do not know at what date the

invention as claimed was made, if ever, I have used the Critical Date of the '915 patent as the point in time for claim interpretation purposes. My opinion does not change if the invention date is earlier.

B. Legal Standards

24. I have been informed by Counsel and understand that documents and materials that qualify as prior art can render a patent claim unpatentable as being anticipated or obvious.

25. I am informed by Counsel and understand that all prior art references are to be looked at from the viewpoint of a person of ordinary skill in the art at the time of the invention, and that this viewpoint prevents one from using his or her own insight or hindsight in deciding whether a claim is anticipated or rendered obvious.

C. Anticipation

26. I understand that patents or printed publications that qualify as prior art can be used to invalidate a patent claim as anticipated or as obvious.

27. I understand that, once the claims of a patent have been properly construed, the second step in determining anticipation of a patent claim requires a comparison of the properly construed claim language to the prior art on a limitation-by-limitation basis.

28. I understand that a prior art reference “anticipates” an asserted claim,

and thus renders the claim invalid, if all limitations of the claim are disclosed in that prior art reference, either explicitly or inherently (i.e., necessarily present).

D. Obviousness

29. I understand that even if a patent is not anticipated, it is still invalid if the differences between the claimed subject matter and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a POSITA.

30. I have been informed by Counsel and understand that a claim is unpatentable for obviousness and that obviousness may be based upon a combination of references. I am informed by Counsel and understand that the combination of familiar elements according to known methods is likely to be obvious when it does no more than yield predictable results. However, I am informed by Counsel and understand 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.

31. I am informed by Counsel and understand that when a patented invention is a combination of known elements, a court determines whether there was an apparent reason to combine the known elements in the fashion claimed by the patent at issue by considering the teachings of prior art references, the effects of demands known to people working in the field or present in the marketplace,

and the background knowledge possessed by a person having ordinary skill in the art.

32. I am informed by Counsel and understand that a patent claim composed of several limitations is not proved obvious merely by demonstrating that each of its limitations was independently known in the prior art. I am informed by Counsel and understand that identifying a reason those elements would be combined can be important because inventions in many instances rely upon building blocks long since uncovered, and claimed discoveries almost of necessity will be combinations of what, in some sense, is already known. I am informed by Counsel and understand that it is improper to use hindsight in an obviousness analysis, and that a patent's claims should not be used as a "roadmap."

33. I am informed by Counsel and understand that an obviousness inquiry requires consideration of the following factors: (1) the scope and content of the prior art, (2) the differences between the prior art and the claims, (3) the level of ordinary skill in the art, and (4) any so called "secondary considerations" of non-obviousness, which include: (i) "long felt need" for the claimed invention, (ii) commercial success attributable to the claimed invention, (iii) unexpected results of the claimed invention, and (iv) "copying" of the claimed invention by others.

34. I have been informed by Counsel and understand that an obviousness evaluation can be based on a single reference or a combination of multiple prior art

references. I understand that the prior art references themselves may provide a suggestion, motivation, or reason to combine, but that the nexus linking two or more prior art references is sometimes simple common sense. I have been informed by Counsel and understand that obviousness analysis recognizes that market demand, rather than scientific literature, often drives innovation, and that a motivation to combine references may be supplied by the direction of the marketplace.

35. I have been informed by Counsel and understand that if a technique has been used to improve one device, and a person of ordinary skill at the time of invention would have recognized that it would improve similar devices in the same way, using the technique is obvious unless its actual application is beyond his or her skill.

36. I have been informed by Counsel and understand that practical and common sense considerations should guide a proper obviousness analysis, because familiar items may have obvious uses beyond their primary purposes. I have been informed by Counsel and understand that a person of ordinary skill looking to overcome a problem will often be able to fit together the teachings of multiple prior art references. I have been informed by Counsel and understand that obviousness analysis therefore takes into account the inferences and creative steps that a person of ordinary skill would have employed at the time of invention.

37. I have been informed by Counsel and understand that a proper obviousness analysis focuses on what was known or obvious to a person of ordinary skill at the time of invention, not just the patentee. Accordingly, I understand that any need or problem known in the field of endeavor at the time of invention and addressed by the patent can provide a reason for combining the elements in the manner claimed.

38. I have been informed by Counsel and understand that a claim can be obvious in light of a single reference, without the need to combine references, if the elements of the claim that are not found explicitly or inherently in the reference can be supplied by the common sense of one of skill in the art.

39. In sum, my understanding is that prior art teachings are properly combined where one of ordinary skill having the understanding and knowledge reflected in the prior art and motivated by the general problem facing the inventor, would have been led to make the combination of elements recited in the claims. Under this analysis, the prior art references themselves, or any need or problem known in the field of endeavor at the time of the invention, can provide a reason for combining the elements of multiple prior art references in the claimed manner.

40. I have been informed by Counsel and understand that in an *inter partes* review, “the petitioner shall have the burden of proving a proposition of unpatentability,” including a proposition of obviousness, “by a preponderance of

the evidence.” 35 U.S.C. §316(e).

VI. MATERIALS CONSIDERED

41. My analysis and conclusions set forth in this declaration are based on my educational background and experiences in the field (*see* Section II). Based on my knowledge and experience, I believe that I am considered to be an expert in the field. Also, based on my knowledge and experience, I understand and know of the capabilities of persons of ordinary skill in the field around the Critical Date.

42. As part of my independent analysis for this declaration, I have considered the following: the background knowledge/technologies that were commonly known to persons of ordinary skill in this art during the time before the Critical Date; my own knowledge and experiences gained from my work experience in the field of the ‘915 patent and related disciplines; and my experience in working with others involved in this field and related disciplines.

43. In addition, I have analyzed the publications and materials listed above.

44. My analysis and conclusions set forth in this declaration are based on the perspective of a POSITA.

VII. THE ‘915 PATENT

A. Overview of the ‘915 Patent

45. The ‘915 patent describes a “Control Center” that provides “automated management of network-based virtual environments.” MS-1001, 6:42-

63. The Control Center computers “manages multiple resources” (e.g., blade servers and other network-based equipment) that “can be configured to support a variety of virtual assets, including desktop computer environments, data storage, application servers, routers, and the like.” *Id.*, 7:5-35. The Control Center also includes modular application components, including an “Asset Manager” and a “Provisioning Manager,” and also includes a “Virtual Mapping engine” that is “programmed to discover the physical resources” and “also the virtual resources, such as virtual servers, storage, desktops, and the like with which the Control Center has access.” *Id.*, 4:4-15, 7:36-59. The Asset Manager provides “visibility and management of all global virtual assets,” including “a global topology view and an inventory of virtual infrastructure from a central console,” and it supports “inventory reports at local, remote, and global context.” *Id.*, 8:7-20.

46. A Provisioning Manager facilitates provisioning through an automated workflow driven by user requests and informed by system inventory. MS-1001, 8:21-45. This includes “an on-demand solution for provisioning of virtual assets for an automated workflow management,” including “allowing users to request and schedule their individual virtual asset needs” and to “prioritize and provision the required assets on demand.” *Id.* Provisioning may also be provided in a portal-driven workflow, for instance, through a “portal for users’ virtual asset requests showing available virtual assets,” and enabling approval of provisioning

processes while “keeping track of used and available assets.” *Id.*

47. The ’915 patent specification’s disclosure is largely focused on an end-to-end, management-plane provisioning workflow within a “control center” rather than as a solution uniquely directed to post-provisioning access. As explained above, the ’915 patent specification includes techniques relating to “a virtual asset provisioning request” and associated sequencing for addressing this request. This sequencing is reinforced by the description of the “Provisioning Manager” as “an on-demand solution for provisioning of virtual assets for an automated workflow management.” *Id.*, 8:21-45.

B. Prosecution History of the ’915 Patent

48. The ’915 patent issued on May 23, 2023 from U.S. Patent App. No. 17/074,500 (“the ’500 Application”). The ’500 Application was filed on October 19, 2020 as a continuation of App. No. 15/201,087, which issued as U.S. Patent No. 10,810,050, which is itself a continuation of App. No. 14/450,765, which issued as U.S. Patent No. 9,444,762. Both of these applications were filed as continuations of App. No. 11/503,090, filed August 10, 2006, and issuing as U.S. Patent No. 8,799,431. The ’500 Application also claims priority to U.S. Provisional Application No. 60/708,473, filed on August 15, 2005. MS-1001, Cover.

49. The prosecution of the ’500 Application included three Office Actions, all of which presented rejections based on Vasilevsky (US 2005/0044301)

("Vasilevsky"). MS-1002, 106-23, 173-86, 230-42, 317-24. Claims 15-19 were withdrawn from consideration throughout prosecution. Claims 1-14 were examined and ultimately allowed.

50. The Examiner issued a first Non-final Office Action on January 18, 2022. MS-1002, 106-23. The Office Action included prior art rejections of claims 1-14 based on Vasilevsky:

Claim Rejections - 35 USC § 102

7. In the event the determination of the status of the application as subject to AIA 35 U.S.C. 102 and 103 (or as subject to pre-AIA 35 U.S.C. 102 and 103) is incorrect, any correction of the statutory basis for the rejection will not be considered a new ground of rejection if the prior art relied upon, and the rationale supporting the rejection, would be the same under either status.

The following is a quotation of the appropriate paragraphs of pre-AIA 35 U.S.C. 102 that form the basis for the rejections under this section made in this Office action:

A person shall be entitled to a patent unless –

(e) the invention was described in (1) an application for patent, published under section 122(b), by another filed in the United States before the invention by the applicant for patent or (2) a patent granted on an application for patent by another filed in the United States before the invention by the applicant for patent, except that an international application filed under the treaty defined in section 351(a) shall have the effects for purposes of this subsection of an application filed in the United States only if the international application designated the United States and was published under Article 21(2) of such treaty in the English language.

Claims 1-14 are rejected under pre-AIA 35 U.S.C. 102(e) as being unpatentable by **Vasilevsky et al. Publication No. (US 2005/0044301 A1)**

MS-1002, 118-19

51. The Applicant filed a Response to Non-Final Office Action on April 18, 2022. MS-1002, 162-72. Applicant amended original claims 1 and 11 to address rejections cited in the first Non-Final Office Action:

1. (Currently Amended) A method for access to one or more virtualization system resources over a network, the method comprising:

implementing a virtualization system that has a control computer and one or more physical ~~resource~~ resources and virtual assets connected to the control ~~control~~ computer, wherein one or more of the virtual assets are capable of being executed by any of the one or more physical resources, and the virtual assets are managed by [[a]] virtualization layer instances running on each of the physical resources; ~~wherein virtual assets are software instantiations of computer systems;~~

connecting an end user to the control computer; and

accessing, through the control computer connection, the virtual assets by the end user.

MS-1002, 163

11. (Currently Amended) A method, comprising:

communicating, by a user, over a network to access one or more virtual assets running in a virtualization system, wherein the virtualization system includes a control computer and one or more physical resources, wherein one or more of the virtual assets are configured to be executed by any of the one or more physical resources, and wherein the virtual assets are managed by virtualization layer instances running on each of the one or more physical resources;

requesting, by the user, provisioning of a virtual asset of the one or more virtual assets in the virtualization system; and

wherein the provisioned virtual asset is a virtual storage and the virtual storage comprises one or more logical storage units mapped to one or more physical storage drives.

MS-1002, 164

The Response also included arguments seeking to distinguish claims 1 and 11 from

the prior art of record:

For instance, Vasilevsky fails to teach or suggest “*implementing a virtualization system that has a control computer and one or more physical resources and virtual assets connected to the control computer, wherein one or more of the virtual assets are capable of being executed by any of the one or more physical resources, and the virtual assets are managed by virtualization layer instances running on each of the physical resources,*” as recited in claim 1 as amended.

Vasilevsky generally describes a system for providing virtual computing services. Particularly, Vasilevsky describes that “a level of abstraction is created between a set of physical processors and a set of virtual multiprocessors to form a virtualized data center.” Vasilevsky, abstract.

For example, in Vasilevsky, a system 101 “may be used to execute one or more data center applications. System 101 may include one or more system layers providing layers of abstraction between programming entities. *As discussed above, a virtualization layer 104 is provided that isolates applications on a guest operating system (GOS) operating in layers 102 and 103, respectively, from an underlying hardware layer 105.*” Vasilevsky, para. [0058]; FIG. 1.

In other words, Vasilevsky describes that one or more physical processors are abstracted from virtual processors, as the “virtualization layer isolates the OS and their applications from the underlying physical nodes.” Vasilevsky, para. [0017].

MS-1002, 169-70

First, Applicant respectfully asserts that Vasilevsky fails to teach or suggest “*wherein one or more of the virtual assets are capable of being executed by any of the one or more physical resources,*” as recited in claim 1 as amended.

As recited in the present specification:

The computer device 300 includes an application that provides a virtualization layer 322 that manages virtual assets 326. Thus, the computer comprises a host machine for the virtualization environment. In FIG. 3, the virtual assets are illustrated as two virtual computers, or machines, a first Virtual Machine 328 and a second Virtual Machine 330. Each virtual machine can include an operating system (OS) for the virtual machine, computer applications programs executing on the virtual machine OS, and virtual hardware assets for the virtual machine.

Spec., para. [0064]; FIG. 3.

In the Office Action, Examiner notes that the “virtual assets” as recited in the present claims are allegedly equivalent to the “operating system layer 103; FIG. 1” in Vasilevsky. Office Action, p. 13. However, in Vasilevsky: “In one aspect, virtualization layer 104 presents, to application layer 102 and operating system layer 103 ***a single system presented in the form of a virtual server.*** In one embodiment, a single instance of an OS is executed by the virtual server.” Vasilevsky, para. [0059]. Accordingly, Vasilevsky describes the operating system layer 103 and application layer 102 as being a single system that is isolated from physical systems 107a-c by virtualization layer 104.

Such a disclosure cannot equate to the virtual assets as recited in the present claims. In the present application, one or more of the virtual assets (e.g., first virtual machine 328, second virtual machine 330) are capable of being executed by any of the one or more physical resources (e.g., computer device 300). Vasilevsky describes that the virtualization layer 104 presents, to application layer 102 and operating system layer 103 a single system presented in the form of a virtual server across all physical systems 107a-c. This disclosure in Vasilevsky makes it impossible for multiple instances of the “single system” to be executed on a single physical processor (e.g., systems 107a-c).

Accordingly, the disclosure of Vasilevsky fails to teach or suggest the virtual assets as recited in the claims.

MS-1002, 170-71

Additionally, Applicant respectfully asserts that Vasilevsky fails to teach or suggest “*the virtual assets are managed by virtualization layer instances running on each of the physical resources,*” as recited in claim 1 as amended.

As recited in the present specification: “The computer device 300 includes an application that provides a virtualization layer 322 that manages virtual assets 326.” Spec., para. [0064]. Accordingly, each physical resource in a network (e.g., 100) can include an application that provides a virtualization layer (e.g., 322) that manages virtual assets (e.g., 326) executing on each physical resource (e.g., computer device 300).

As discussed above, Vasilevsky describes a virtualization layer 104 that isolates applications on a guest operating system (GOS) operating in layers 102 and 103, respectively, from an underlying hardware layer 105. In other words, the virtualization layer in Vasilevsky does not execute each of the physical resources (e.g., systems 107a-c) but rather, is a single virtualization layer instance running in a system (e.g., 100). Vasilevsky describes that the virtualization layer 104 presents, to application layer 102 and operating system layer 103 a single system presented in the form of a virtual server across all physical systems 107a-c. Accordingly, Vasilevsky’s disclosure of a virtualization layer to present a single system does not equate to the virtualization layer instances running on each of the physical resources, as recited in claim 1 as amended.

MS-1002, 171

52. The Examiner issued a second non-final Office Action on May 12, 2022. This Office Action also included prior art rejections based on Vasilevsky:

Claim Rejections - 35 USC § 102

8. In the event the determination of the status of the application as subject to AIA 35 U.S.C. 102 and 103 (or as subject to pre-AIA 35 U.S.C. 102 and 103) is incorrect, any correction of the statutory basis for the rejection will not be considered a new ground of rejection if the prior art relied upon, and the rationale supporting the rejection, would be the same under either status.

The following is a quotation of the appropriate paragraphs of pre-AIA 35 U.S.C. 102 that form the basis for the rejections under this section made in this Office action:

A person shall be entitled to a patent unless –

(e) the invention was described in (1) an application for patent, published under section 122(b), by another filed in the United States before the invention by the applicant for patent or (2) a patent granted on an application for patent by another filed in the United States before the invention by the applicant for patent, except that an international application filed under the treaty defined in section 351(a) shall have the effects for purposes of this subsection of an application filed in the United States only if the international application designated the United States and was published under Article 21(2) of such treaty in the English language.

Claims 1-14 are rejected under pre-AIA 35 U.S.C. 102(e) as being unpatentable by **Vasilevsky et al. Publication No. (US 2005/0044301 A1)**.

MS-1002, 180

53. The Applicant filed a Response to Office Action on January 7, 2019.
MS-1002, 185-91. Applicant further amended each of claims 1 and 11:

1. (Currently Amended) A method for access to one or more virtualization system resources over a network, the method comprising:

implementing a virtualization system that has a control computer and one or more physical resources and virtual assets connected to the control computer, ~~wherein one or more of the virtual assets are capable of being executed by any of the one or more physical resources,~~ wherein each of the one or more physical resources execute at least two virtual assets that each comprise software instantiations of computer systems, and the virtual assets are managed by virtualization layer instances running on each of the physical resources;

connecting an end user to the control computer; and

accessing, through the control computer connection, the virtual assets by the end user.

MS-1002, 213

11. (Currently Amended) A method, comprising:

communicating, by a user, over a network to access one or more virtual assets running in a virtualization system, wherein the virtualization system includes a control computer and one or more physical resources, ~~wherein one or more of the virtual assets are configured to be executed by any of the one or more physical resources,~~ wherein each of the one or more physical resources execute at least two virtual assets that each comprise software instantiations of computer systems, and wherein the virtual assets are managed by virtualization layer instances running on each of the one or more physical resources;

MS-1002, 214

The Response also included arguments seeking to distinguish claims 1 and 11 from the prior art of record:

First, Applicant submits that Vasilevsky fails to teach or suggest “*wherein each of the one or more physical resources execute at least two virtual assets that each comprise software instantiations of computer systems,*” as recited in claim 1 as amended.

Vasilevsky generally describes a system for providing virtual computing services. Particularly, Vasilevsky describes that “a level of abstraction is created between a set of physical processors and a set of virtual multiprocessors to form a virtualized data center.” Vasilevsky, abstract.

Further, Vasilevsky describes a virtualization layer 104 that isolates applications on a guest operating system (GOS) operating in layers 102 and 103, respectively, from an underlying hardware layer 105. In other words, the virtualization layer in Vasilevsky virtualizes each of the physical resources (e.g., systems 107a-c) to implement a single computing system (e.g., system 101) independent of any individual system 107a-c. Accordingly, it is clear that a single “system 107a-c” as described in Vasilevsky is not independently capable of implementing a virtual machine. Rather, each individual “system e.g., 107a-c” as described in Vasilevsky is virtualized with other systems 107a-c by a virtualization layer to implement a single virtualization system 101.

In contrast, claim 1 as amended recites “*one or more physical resources execute at least two virtual assets that each comprise software instantiations of computer systems.*”

As noted above, a “system 107a-c” as described in Vasilevsky is not equivalent to a “physical resource” as recited in the present claims. For instance, the “system 107a-c” as described in Vasilevsky is not disclosed to “*execute at least two virtual assets that each comprise software instantiations of computer systems,*” as recited in claim 1 as amended.

Accordingly, the disclosure of Vasilevsky fails to teach or suggest “wherein each of the one or more physical resources execute at least two virtual assets that each comprise software instantiations of computer systems,” as recited in claim 1 as amended.

Further, Vasilevsky describes a virtualization layer 104 that isolates applications on a guest operating system (GOS) operating in layers 102 and 103, respectively, from an underlying hardware layer 105. In other words, the virtualization layer in Vasilevsky virtualizes each of the physical resources (e.g., systems 107a-c) to implement a single computing system (e.g., system 101) independent of any individual system 107a-c. Accordingly, it is clear that a single “system 107a-c” as described in Vasilevsky is not independently capable of implementing a virtual machine. Rather, each individual “system e.g., 107a-c” as described in Vasilevsky is virtualized with other systems 107a-c by a virtualization layer to implement a single virtualization system 101.

In contrast, claim 1 as amended recites “*one or more physical resources execute at least two virtual assets that each comprise software instantiations of computer systems.*”

As noted above, a “system 107a-c” as described in Vasilevsky is not equivalent to a “physical resource” as recited in the present claims. For instance, the “system 107a-c” as described in Vasilevsky is not disclosed to “*execute at least two virtual assets that each comprise software instantiations of computer systems,*” as recited in claim 1 as amended.

Accordingly, the disclosure of Vasilevsky fails to teach or suggest “wherein each of the one or more physical resources execute at least two virtual assets that each comprise software instantiations of computer systems,” as recited in claim 1 as amended.

Additionally, Applicant respectfully asserts that Vasilevsky fails to teach or suggest “*the virtual assets are managed by virtualization layer instances running on each of the physical resources,*” as recited in claim 1 as amended.

As recited in the present specification: “The computer device 300 includes an application that provides a virtualization layer 322 that manages virtual assets 326.” Spec., para. [0064]. Accordingly, each physical resource in a network (e.g., 100) can include an application that provides a virtualization layer (e.g., 322) that manages virtual assets (e.g., 326) executing on each physical resource (e.g., computer device 300).

As discussed above, Vasilevsky describes a virtualization layer 104 that isolates applications on a guest operating system (GOS) operating in layers 102 and 103, respectively, from an underlying hardware layer 105. In other words, the virtualization layer in Vasilevsky does not execute each of the physical resources (e.g., systems 107a-c) but rather, is a single virtualization layer instance running in a system (e.g., 100). Vasilevsky describes that the virtualization layer 104 presents, to application layer 102 and operating system layer 103 a single system presented in the form of a virtual server across all physical systems 107a-c. Accordingly, Vasilevsky's disclosure of a virtualization layer to present a single system does not equate to the virtualization layer instances running on each of the physical resources, as recited in claim 1 as amended.

MS-1002, 217-18

54. Following this response, the Examiner issued another Office Action on September 9, 2022. MS-1002, 230-42. In this Office Action, the Examiner issued similar prior art rejections of claims 1-14 based on Vasilevsky and U.S. Pat. App. Pub. No. 2006/0069761 ("Singh"):

Claim Rejections - 35 USC § 103

9. In the event the determination of the status of the application as subject to AIA 35 U.S.C. 102 and 103 (or as subject to pre-AIA 35 U.S.C. 102 and 103) is incorrect, any

correction of the statutory basis for the rejection will not be considered a new ground of rejection if the prior art relied upon, and the rationale supporting the rejection, would be the same under either status.

The following is a quotation of pre-AIA 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

(a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negated by the manner in which the invention was made.

The factual inquiries for establishing a background for determining obviousness under pre-AIA 35 U.S.C. 103(a) are summarized as follows:

1. Determining the scope and contents of the prior art.
2. Ascertaining the differences between the prior art and the claims at issue.
3. Resolving the level of ordinary skill in the pertinent art.
4. Considering objective evidence present in the application indicating obviousness or nonobviousness.

Claims 1-14 are rejected under pre-AIA 35 U.S.C. 103(a) as being unpatentable over **Vasilevsky et al. Publication No. (US 2005/0044301 A1)** in view of **Singh et al. Publication No. (US 2006/0069761 A1)**.

MS-1002, 235-36

55. Applicant then submitted another Response on December 9, 2022. In this Response, Applicant did not amend any claims but included the following remarks attempting to distinguish the prior art of record:

I. None of the cited references teach or suggest *the virtual assets are managed by virtualization layer instances running on each of the physical resources*, as recited in claim 1

First, Applicant respectfully submits that the cited references fail to teach or suggest “*the virtual assets are managed by virtualization layer instances running on each of the physical resources*,” as recited in claim 1.

Applicant respectfully disagrees with Examiner that the first reference, Vasilevsky, discloses this limitation.

Vasilevsky generally relates to a grid computing system that combines resources of multiple servers to create a *single* virtualized data center. Further, Vasilevsky describes a virtualization layer 104 that isolates applications on a guest operating system (GOS) operating in layers 102 and 103, respectively, from an underlying hardware layer 105. This virtualization layer 104 virtualizes resources on each individual system 107a-c system to implement a single computing system (e.g., system 101) that is independent of any individual system 107a-c.

In contrast, the present claims recite that *the virtual assets are managed by virtualization layer instances running on each of the physical resources*. See Spec., para. [0064]; FIG. 3 (“The computer device 300 includes an application that provides a virtualization layer 322 that manages virtual assets 326.”). Further, claim 1 recites that *each of the one or more physical resources execute at least two virtual assets that each comprise software instantiations of computer systems*. See Spec., para. [0064]; FIG. 3 (“Thus, the computer comprises a host machine for the virtualization environment. In FIG. 3, the virtual assets are illustrated as two virtual computers, or machines, a first Virtual Machine 328 and a second Virtual Machine 330.”).

Accordingly, Vasilevsky’s discussion of a single virtualization layer that virtualizes resources on each system 107a-c to implement a single computing system (e.g., system 101) clearly does not teach or suggest the *virtual assets are managed by virtualization layer instances running on each of the physical resources*, as recited in claim 1.

MS-1002, 280-81

Singh fails to cure the deficiencies of Vasilevsky. Singh generally relates to technique[s] for load balancing among the servers of a computer network. Singh also generally describes an ability that a “server of the computer network may host one or more virtual machines.” Singh, para. [0012].

However, Singh explicitly fails to teach or suggest *the virtual assets are managed by virtualization layer instances running on each of the physical resources*, as recited in claim 1. Singh at para. [0013] states “Virtualization layer 20 includes the hardware of the server, the operating system of the server, and the virtual system software that runs on the operating system of the server and supports each virtual machine of the server.” In other words, Singh requires that the server runs an operating system, and that the virtual system software runs on the operating system. Such a disclosure cannot equate to *the virtual assets are managed by virtualization layer instances running on each of the physical resources*, as recited in claim 1 as amended.

Accordingly, Singh cannot teach or suggest the virtual assets being managed by virtualization layer instances running on each of the physical resources, as recited in claim 1.

Accordingly, none of the cited references teach or suggest *the virtual assets are managed by virtualization layer instances running on each of the physical resources*, as recited in claim 1.

MS-1002, 281-82

56. The Applicant also requested a telephonic interview on December 7, 2022 and no agreement was reached. MS-1002, 291. Following this interview and Applicant’s response, the Examiner issued a Notice of Allowance on January 23, 2023. *Id.*, 317-23. The Examiner identified the following reasons for allowance in the Notice of Allowance:

REASONS FOR ALLOWANCE

2. The following is an examiner's statement of reasons for allowance:

There is no teaching or suggestion in the prior art of the claimed limitations, as presented in the independent claims, specifically employing a virtualization system that has a control computer and one or more physical resources and virtual assets connected to the control computer, each of the one or more physical resources execute at least two virtual assets that each comprise software instantiations of computer systems, and the virtual assets are managed by virtualization layer instances running on each of the physical resources, connecting an end user to the control computer and accessing the virtual assets by the end user through the control computer connection as detailed in the dependent claims for such limitations.

MS-1002, 322

C. Claim Interpretation

57. In my opinion, the grounds discussed in Section X render the Challenged Claims obvious under any reasonable interpretation of the claim language, including plain and ordinary meaning. The analysis set forth in this declaration applies the following constructions, which are consistent with disclosure found in the '915 patent, and therefore, demonstrate subject matter that falls within the meaning of these terms. The prior art analysis included below in Section X are not inconsistent with these constructions.

Claim Term(s)	Proposed Construction
“control computer”	“A computing device that is separate and distinct from the physical host machines being managed and is not itself deployed as a virtual asset.” MS-1001, 5:66-6:35, 7:5-17.
“virtual assets”	“Software instantiations of equipment objects that exist only within a virtual environment” MS-1001, 6:36-41.
“resources”	“Physical devices on the network, such as computers, printers, routers, and other tangible hardware” MS-1001, 6:36-41.
“physical resources”	“Physical devices on the network, such as computers, printers, routers, and other tangible hardware” MS-1001, 6:36-41.
“network resource”	“Physical network infrastructure devices, including routers, switches, bridges, infiniband, wireless, radio, optical, and fiber channel equipment” MS-1001, 2:14-30.
“software instantiations of computer systems”	“Software that can virtually perform functions of physical devices that can connect to computers”

	MS-1001, 3:13-24, 10:1-23.
“provisioning” / “provisioned”	“Associating a virtual asset to a particular physical computer resource” MS-1001, 20:19-39.
“control computer connection”	“A communication pathway where the control computer acts as an intermediary or gateway in the data path, such that the end user’s access to the virtual assets must be routed directly through the control computer itself” MS-1001, 5:66-6:35, 6:64-7:4, 22:10-28
“logical”	“A software-defined abstraction that requires direct mapping to underlying physical hardware” MS-1001, 20:45-21:3.
“physical storage drives”	“Specific, individual hardware components (e.g., a specific hard disk or a physical router) that are directly addressed and mapped to by the virtualization layer” MS-1001, 20:45-21:3.

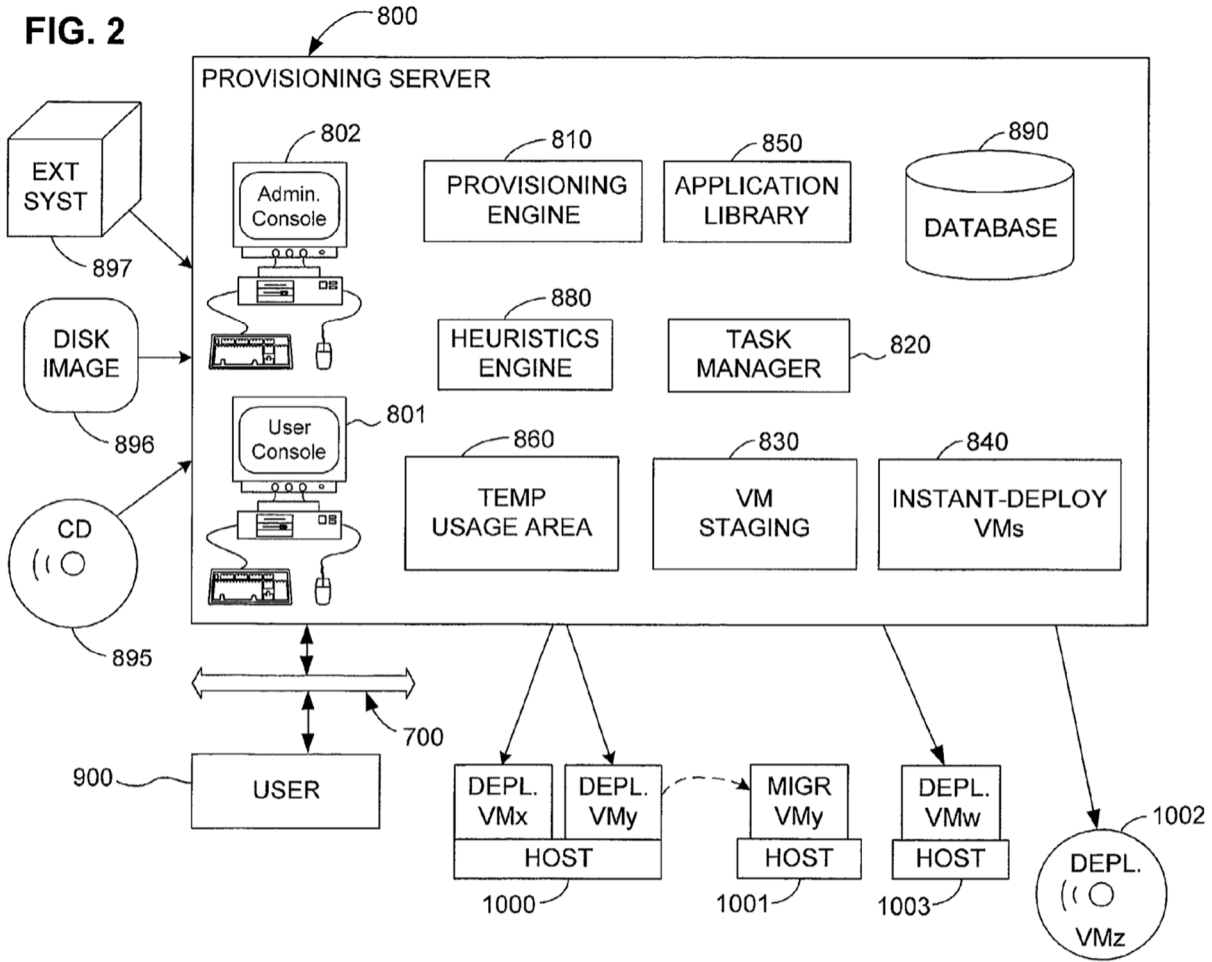
VIII. OVERVIEW OF THE PRIOR ART

A. Khandekar (MS-1004)

58. Khandekar is focused on a VM provisioning architecture in which a user requests a computer having desired configuration characteristics through a centralized provisioning server. MS-1004, 9:65-10:4. Specifically, Khandekar

describes that “the user console is preferably a web server,” and the user “interacts with [it] with any conventional web browser,” where the user can “custom-provision a VM” or select a “pre-configured, instantly deployable VM.” MS-1004, 13:66-14:3. Khandekar describes that a user submits configuration selections (e.g., OS configuration such as “hostname, IP address, MAC address”) and selects desired applications, and then “submits the request to the provisioning server,” which applies heuristics/rules to select an appropriate staged VM and an appropriate host on which to deploy that VM. MS-1004, 9:65-10:42.

59. Figure 2 shows an example of a system 800 used for provisioning:



MS-1004, FIG. 2

60. As shown above, Khandekar's centralized VM provisioning environment allows one or more users 900 to communicate with the provisioning server 800 via a network 700. MS-1004, 9:46-64. The VM provisioning server (e.g., 800) includes a front-end “user console 801” used to “present the various options to the user and to submit the selected options,” and may also include an “administration console 802” for “monitor[ing] the operational state” and “maintain[ing] the database.” *Id.*, 9:46-64, 10:15-26. The provisioning server

receives user input identifying desired configuration characteristics (e.g., operating system selections, application selections, and “OS configuration, for example, hostname, [and] IP address”) and applies a rules/heuristics component (e.g., heuristics engine 880) to “select the most appropriate VM from a library 830 of staged VMs” and to “select[] which host machine 1000, 1001 is most appropriate to provision and deploy the VM on.” *Id.*, 9:65-10:42.

61. Khandekar's host systems provide an execution platform for deployed VMs (e.g., VMx and VMy). Khandekar further describes that "any number of VMs may be deployed and loaded on a single host," with VMs also capable of being migrated or copied between hosts (e.g., "virtual machine VMy is shown as having been migrated (or copied) from host 1000 to host 1001"). MS-1004, 10:50-60, 12:57-13:8. Workflow components also support provisioning sessions, including a temporary usage area 860 that "holds the user input as well as the various temporary data needed while the provisioning task is in progress." MS-1004, 10:61-67, 11:1-10. A task manager 820 also initiates a “transaction” and “returns a task identifier” that can be used to "find out the progress of the task.” *Id.*

62. Once the provisioning server provisions a VM on the selected host, Khandekar describes that “the user can then access the newly created VM via any known remote access software,” including VMware remote console software. MS-1004, 14:4-18. Khandekar also contemplates provisioning requests submitted

programmatically, explaining that “software applications acting as a user interact with the provisioning server using a well-defined programmatic interface (API).” *Id.*, 9:46-64.

63. Khandekar is analogous prior art to the ’915 patent because it is in the same general field of endeavor, e.g., management of computer network virtualization environments. The ’915 patent describes centralized management through a Control Center providing functions, such as building an inventory of physical resources and virtual assets, provisioning virtual assets, and managing the virtualization environment across platforms. Khandekar also addresses centralized VM provisioning in which users interact with a provisioning server via a user console and provision VMs (including “instantly deployable” VMs). *See* MS-1004, 9:65-10:14, 13:66-14:3. Khandekar also describes monitoring deployed VMs and taking management actions, such as migrating VMs based on host load, which reflects similar operational management concerns that would have been important to a POSITA working on the problems addressed in the ’915 patent. *See id.*, 13:44-56.

B. Hirschfeld (MS-1006)

64. Hirschfeld describes a “virtualized logical server cloud” in which users access logical servers through a portal-like user interface without needing to know the access information of the underlying physical servers. MS-1006, 4:37-52,

6:64-7:30, 17:25-39. Although virtualization allows “multiple logical servers to override a physical server's attributes and share its resources” (*id.*, 2:1-7) users historically “had to be aware of and have specific information about the underlying physical server to access the logical servers” (*id.*, 2:8-35), such as the physical server's IP address/server name and port number, and incorrect access inputs could result in access being denied. MS-1006, 4:37-52, 6:64-7:30, 17:25-39.

65. Hirschfeld identifies a desire “to completely separate logical servers from physical servers so that there is no permanent tie between physical and logical resources,” which enables physical servers to function as a resource pool that supports logical servers and allowing a logical server to be reallocated to a different physical server “without users experiencing any change in access approach.” MS-1006, 2:50-59.

66. Hirschfeld also describes a server cloud manager (SCM) that provides a “web page portal or other access mechanism” for remote terminals to communicate with the logical server cloud, such that “a user of any remote terminal” may “login to and access any of the logical servers” via the user interface and SCM. MS-1006, 6:28-63. A user may employ a “local browser to access a web page” associated with the user interface, and the interface provides a “portal or other interface” through which the user may access one or more logical servers based on permission (e.g., by “point and click” selection of a logical server

graphic). *Id.*, 6:64-7:30.

67. Figure 1 shows an overview of a virtualized logical server cloud implemented with a server cloud manager:

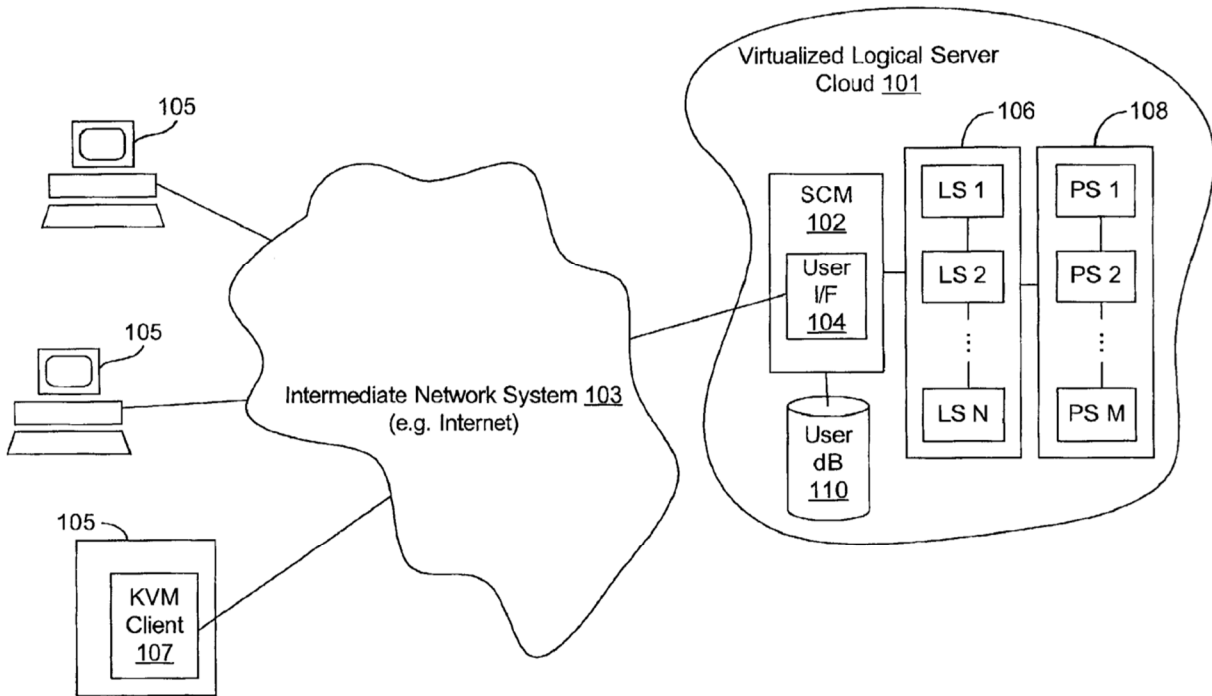


FIG. 1

MS-1006, FIG. 1

68. As shown above, Hirschfeld's virtualized logical server cloud 101 allows remote terminals 105 to access the system via an intermediate network system 103. MS-1006, 6:28-63. The environment includes one or more "logical servers 106" and a "bank of physical servers 108," where each logical server "is linked to and implemented on a corresponding one of the physical servers 108." *Id.* This architecture includes a server cloud manager (SCM) 102, which "interfaces

and controls each of the logical servers 106 and the bank of physical servers 108.”

Id. The SCM 102 includes a “user interface (I/F) 104” that provides a “web page portal or other access mechanism” allowing a user to “login to and access any of the logical servers 106.” *Id.*

69. To manage the separation of logical and physical resources, the SCM 102 utilizes a “user database 110” that “incorporates a mapping of users, the logical servers 106 and the physical servers 108.” *Id.*, 7:32-48. This database maintains the “link information that maps each logical server 106 with one of the physical servers 108,” enabling the SCM to automatically access the appropriate physical server without the user needing to supply or be aware of physical server information. *Id.*, 7:32-62.

70. Hirschfeld also describes a user database that “incorporates a mapping of users, the logical servers and the physical servers,” including “link information that maps each logical server” with a corresponding physical server, such that the SCM “automatically accesses the appropriate physical server corresponding to the selected logical server,” and the user “does not have to supply physical server information” and “does not even need to be aware of” physical-server access/location information. MS-1006, 7:32-62. Hirschfeld emphasizes that its virtualization “enables a logical Server to exist independent of a specific physical server,” with logical attributes “non-deterministically allocated to physical

resources,” which creates a cloud of logical servers over one or more physical servers. *Id.*, 17:9-24.

71. Figure 2 illustrates the relationships between the logical servers forming a logical server cloud, the underlying bank of physical servers, and the server cloud manager.

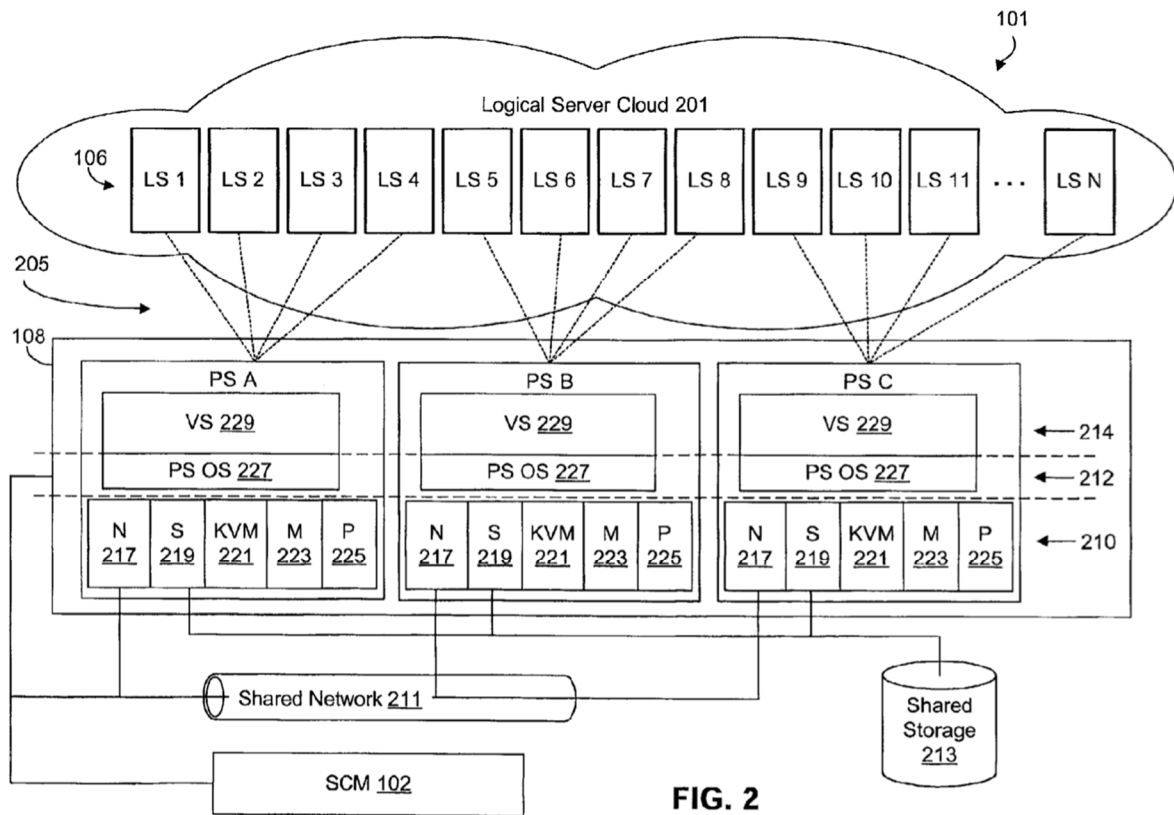


FIG. 2

EX-1006, FIG. 2

72. As shown above, Hirschfeld depicts a “logical server cloud 201” containing multiple “logical servers 106” that operate above a “bank of physical servers 108.” MS-1006, 7:63-8:10. These layers are decoupled, such that “each of the logical servers 106 is non-deterministically linked to a corresponding physical

server 108 as illustrated by a corresponding links 205 (illustrated by dashed lines).” *Id.*, 8:1-10. This allows a logical server to be moved to a different physical server “without destroying the identity of the moved logical server 106 as determined by its ‘persistent attributes’.” *Id.*, 8:11-38. The underlying physical servers 108 provide the physical resources, including a “hardware layer 210” (which incorporates a “memory resource ‘M’ 223” and a “processor resource ‘P’ 225”) and an “operating system (OS) layer 212.” *Id.*, 8:39-64. By maintaining the logical servers 106 in a separate cloud 201 connected only by the dashed links 205, Figure 2 visually demonstrates how the system manages logical servers independently from the specific hardware and operating systems of the underlying physical servers. *Id.*, 7:49-62, FIG. 2.

73. Hirschfeld is analogous prior art to the ’915 patent because it is in the same general field of endeavor, e.g., virtualization technology and centralized management of virtualized servers. The ’915 patent generally relates to management of computer network virtualization environments. Hirschfeld similarly describes a server cloud manager (SCM) for a virtualized logical server cloud. *See* MS-1006, 11:45-57, 11:58-12:19. Hirschfeld also addresses problems aligned with scalable virtualization management (e.g., ensuring user access is substantially unchanged when a logical server is reallocated to different physical servers) and describes SCM functionality to match demand. *See* MS-1006, 17:41-

49, 18:1-17.

C. Bandhole (MS-1007)

74. Bandhole describes a remotely administered “dynamic computing environment” (DCE) framework that is focused on “configuring, provisioning, accessing and managing” such environments through user-facing and programmatic interfaces. MS-1007, [0032]. Bandhole explains that its administrative interfaces “can be used interactively through a graphical user interface such as a web page or non-interactively through a program script,” and that the framework permits “‘virtual’ hosting” in which users can “remotely configure, provision, access, and manage DCEs” from “any geographic location.” *Id.*, [0032], [0033].

75. Figure 3 shows the component architecture of a system configured to host and administer these dynamic computing environments:

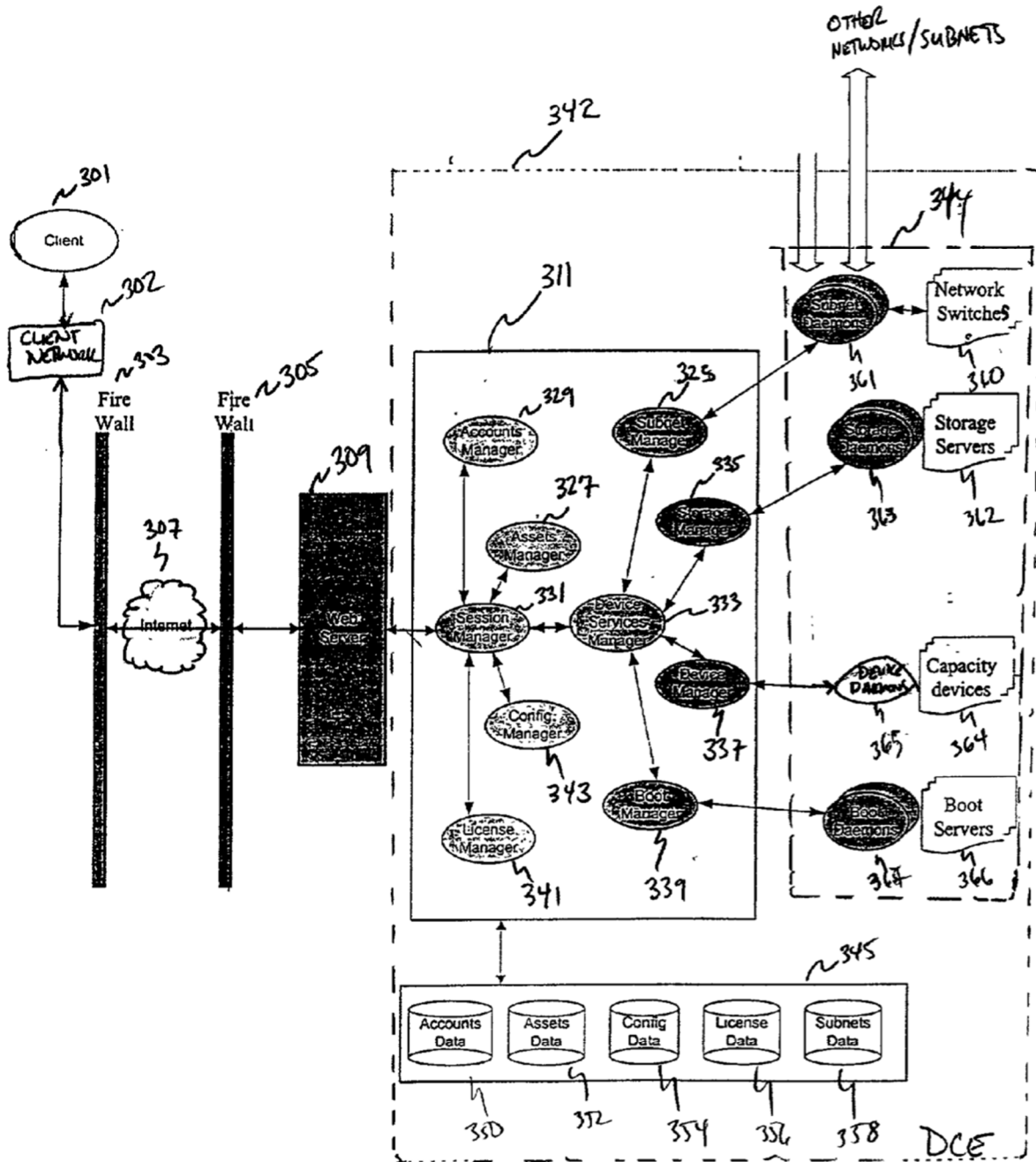


FIG. 3

MS-1007, FIG. 3

76. As shown above, Bandhole's client/server architecture allows a client (e.g., client 301) to communicate with the system over a network such as the Internet (e.g., Internet 307). MS-1007, [0047]. The environment accommodates client computing devices that "can be made in many dimensions and styles (e.g., laptop, palmtop, server, workstation, mainframe)," enabling users to interact with the system using known hardware platforms. *Id.*, [0035].

77. Within this architecture, the web server 309 uses "Java Servlets and JSP" to "receive requests from any web client, such as a browser," and "forward such requests to the application server 311." *Id.*, [0048]. To manage the dynamic computing environment, the application server 311 operates to present the user with "methods to create a DCE," "methods to allocate physical resources required for a DCE," and methods to "manage and to interact with a DCE as well as its components." *Id.*, [0051].

78. Bandhole is analogous prior art to the '915 patent because it is in the same general field of endeavor, e.g., remotely administered virtual computing environments implemented using physical resources and managed through centralized software services. Bandhole similarly describes a framework for configuring, provisioning, accessing, and managing dynamic computing environments (DCEs) remotely. *See* MS-1007, [0015], [0016], [0032], [0033]. Bandhole also recognizes that conventional infrastructures impose undue costs and

efforts to provision or maintain resources and that the burden increases non-linearly with complexity. *See id.*

D. Vasilevsky (MS-1005)

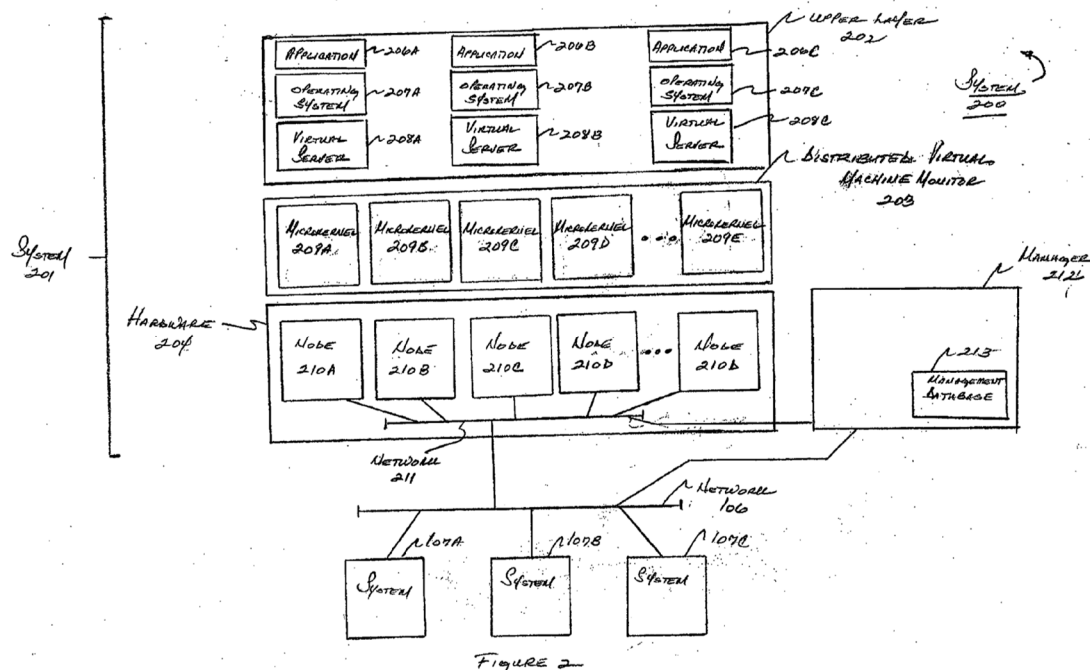
79. Vasilevsky describes approaches to virtualization management directed to dynamic (i.e., runtime) provisioning and reconfiguration of compute resources in a virtual computing system. MS-1005, [0150], [0179]. In particular, Vasilevsky replaces static resource allocation with a “capability for dynamically reconfiguring resources” such that “resources may be allocated (or deallocated) transparently to the applications.” *Id.*, [0179].

80. To enable runtime control, Vasilevsky describes a management architecture that allows configuration changes among “multiple entities (administrators, programs, etc.)” and provides “capability for dynamically reconfiguring resources” such that “resources may be allocated (or deallocated) transparently to the applications.” MS-1005, [0084], [0179]. The management agent maintains a “mapping database that defines mapping of virtual resources to physical hardware,” including mapping a “virtual processor” to a “physical processor,” and, in some implementations, virtual processors and partitions that can span physical nodes. *Id.*, [0091], [0094], [0150], [0179].

81. Vasilevsky further explains that such capabilities may be implemented via a management server/agent arrangement in which the management server

writes configuration information (e.g., indicating how each node should be configured) to a data store that management agents read to configure distributed server/virtual server execution. MS-1005, [0085], [0088], [0155]. For resource assignment, a manager may be adapted to perform an association between "unallocated actual resources" and virtual servers. *Id.*, Cl. 84. The system maintains a "mapping database that defines mapping of virtual resources to physical hardware" and defines virtual resources backed by assigned actual resources (e.g., mapping a "virtual network interface" to an "actual network interface"). *Id.*, [0150].

82. Figure 2 illustrates an exemplary system architecture for providing virtual services, including a manager that monitors system performance.



MS-1005, FIG. 2

83. As shown above, Vasilevsky's system 201 includes an upper layer 202 with virtual servers 208A-208C, an abstraction layer provided by a distributed virtual machine monitor (DVMM) 203, and an underlying hardware layer 204. MS-1005, [0063]-[0066]. To oversee this environment, the system includes a “manager 212 that manages the configuration of system 201.” *Id.*, [0070]. Manager 212 communicates with a management agent executing on the virtual servers “for the purpose of performing configuration changes, monitoring performance, and performing other administrative functions.” *Id.*, [0070].

84. Vasilevsky is analogous prior art to the '915 patent because it is in the same general field of endeavor, e.g., networked computer systems used to provide resources in a data center using virtualization and management infrastructure. The '915 patent is directed to management of computer network virtualization environments. Vasilevsky addresses similar technical problems identified in the '915 patent specification, including that conventional data center configuration changes are often performed manually and can require interruption or reboot. *See* MS-1005, [0004], [0075]-[0079]. Vasilevsky also describes a management architecture employing a manager and management agents (including use of a data store) to control configuration of virtual/distributed servers, which is the same type of virtualization-environment management infrastructure a POSITA would have consulted when addressing the '915 patent's centralized management

objectives. *See id.*, [0071]-[0074], [0080].

E. Fujino (MS-1008)

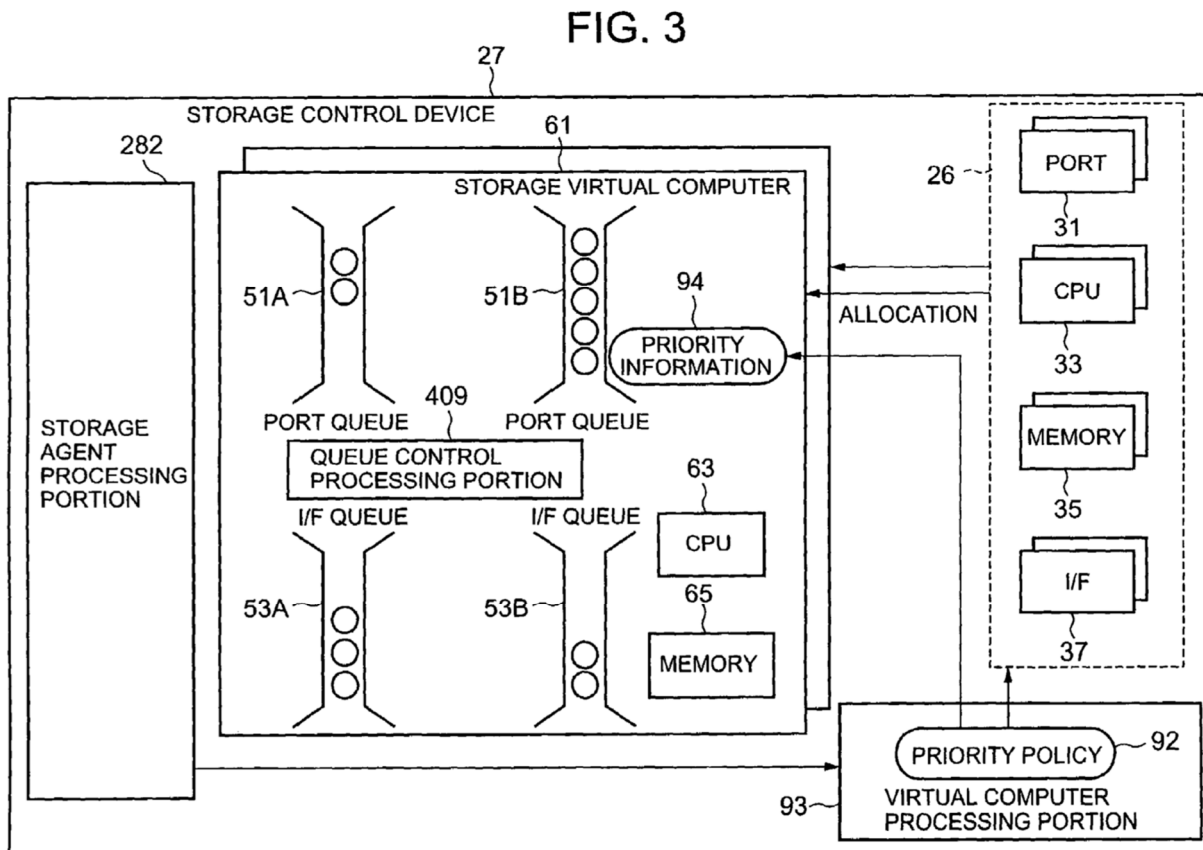
85. Fujino describes virtual computer instantiation and resource pool allocation/monitoring in a storage-control context, including agent-driven creation/deletion of virtual computers and tracking how pooled resources are allocated. Specifically, a “storage agent processing portion” is “capable of monitoring the state of the resource pool,” e.g., “the usage proportion of the CPUs, the usable capacity of the memory” and the number of existing “storage virtual computers.” MS-1008, 8:47-9:3. Based on the monitoring result, this causes the “virtual computer processing portion” to “generate or delete” the storage virtual computer. *Id.*

86. The virtual computer processing portion is also capable of generating or deleting a virtual computer “allocated all or a part of” the resource pool, and further includes “an allocated CPU resource” as a “virtual CPU” and an “allocated memory resource” as a “virtual memory.” *Id.*, 9:4-29. It also includes queueing structures (e.g., port queue, which is “capable of storing information elements”). *Id.* Fujino also discloses resource-management operations performed upon receiving requests (e.g., copy/move/modify logical volumes, including “modify[ing] the storage capacity of the logical volume”). *See id.*, 26:56-27:12.

87. Fujino describes virtual computer instantiation, storage virtualization,

and resource pool monitoring in a storage-control context. MS-1008, 8:47-9:29.

Figure 3 illustrates the architecture of a storage control device providing these capabilities:



MS-1008, FIG. 3

88. As shown above, Fujino provides a storage virtualization architecture in which a storage control device 27 controls physical hardware, including “storages (hard disk drives, for example) 39” and physical network connections such as “ports 31.” MS-1008, 6:40-49, 9:4-29. To manage these resources, Fujino utilizes a “storage agent processing portion 282” that is “capable of monitoring the

state of the resource pool 26,” including tracking “which part of the resource pool 26 is allocated to which virtual computer.” *Id.*, 8:47-9:3. Based on these monitoring results, the agent can cause a “virtual computer processing portion 93” to “generate or delete” a “storage virtual computer 61.” *Id.*

89. When generated, the storage virtual computer 61 is “allocated all or a part of” the resource pool, including “an allocated CPU resource” (virtual CPU 63) and “an allocated memory resource” (virtual memory 65). *Id.*, 9:4-29. Importantly, the storage virtual computer 61 is configured with logical, port-side interface constructs, such as a “virtual port queue 51” that is “capable of storing information elements” to be “input or output into/from the port 31.” *Id.* Through this interface architecture, the storage virtual computer 61 performs I/O by receiving data through the physical port 31 and “writ[ing] the write subject data” into a “logical volume 41” backed by the underlying hard disk drive storages 39. *Id.*, 14:15-48.

90. Fujino is analogous prior art to the '915 patent because it is in the same general field of endeavor, e.g., centralized management of virtualized computing resources in networked enterprise environments. The '915 patent generally relates to the management of computer network virtualization environments. Fujino similarly describes a management device that monitors hardware resource usage across storage control devices and configures a storage virtual computer using unused resources to handle access requests. *See* MS-1008,

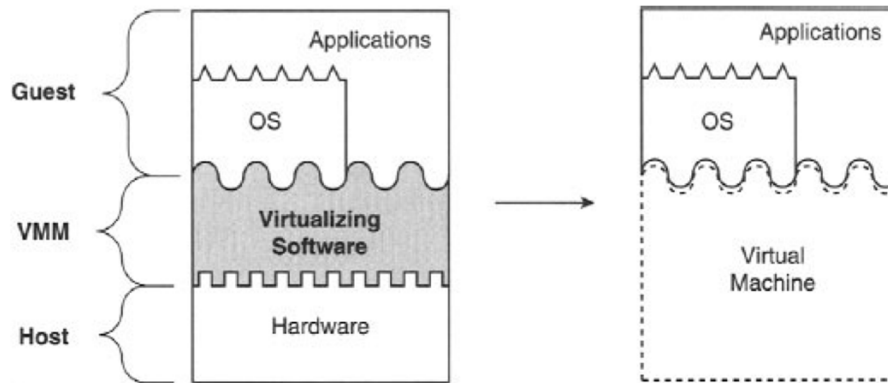
1:30-44. Fujino also addresses similar technical problems to those identified in the '915 patent specification, such effective control and management of complex, resource-constrained infrastructure. *See* MS-1008, 1:57-2:32.

F. Smith (MS-1010)

91. Smith is a book published in June 2005 that describes virtual machine technology and explains that such technology “enable[s] new capabilities and to solve a variety of problems in interfacing major computer system components.” MS-1010, xi. Smith explains that virtualization allows a real system to appear as a different virtual system or as multiple virtual systems, thereby providing a framework in which multiple execution environments may coexist on shared underlying hardware. *Id.*, 3, 9.

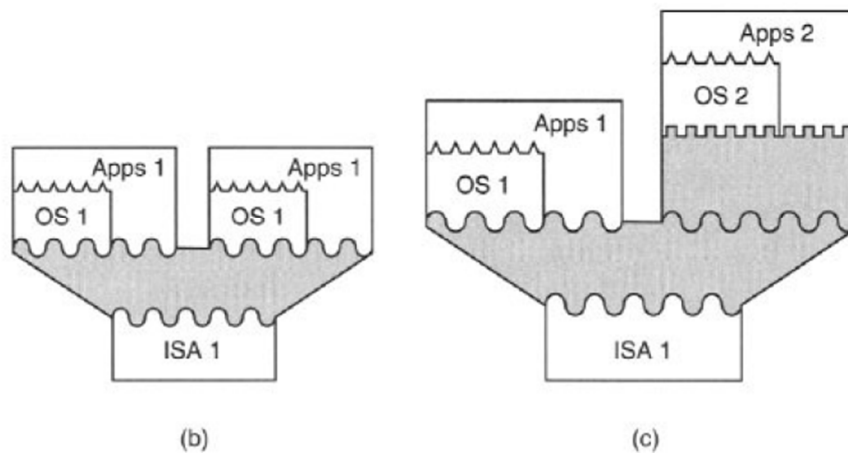
92. Smith describes a “system VM environment” in which the real resources of a host platform are shared among guest system virtual machines through a layer of software called a virtual machine monitor, or VMM. MS-1010, 369. Smith explains that the VMM “manag[es] the allocation of, and access to, the hardware resources of the host platform,” that the VMM “owns the real system resources and makes them available to one or more guest operating systems that alternately execute on the same hardware,” and that each guest operating system is given the illusion of owning the resources it allocates to its user programs. *Id.* Smith also explains that a VM is implemented by adding a layer of software to a

real machine to support the desired virtual-machine architecture, and that such virtualizing software is often referred to as the VMM. *Id.*, 4, 11.



MS-1010, FIG. 1.7

93. Smith further describes that “[m]ultiple, replicated virtual machines can be implemented on a single hardware platform to provide individuals or user groups with their own operating system environments.” MS-1010, 5. Smith illustrates examples in which separate VMs on the same hardware platform may run the same operating system or different operating systems, while supporting different application sets in those separate VM environments. *Id.*, Figures 1.8(b)-(c). A POSITA would have understood from these teachings that Smith describes a conventional virtualization architecture in which multiple software-implemented VM environments operate concurrently on shared host hardware under VMM control.



MS-1010, Figures 1.8 (b)-(c)

94. Smith also describes that virtualization provides isolation, security, and robustness benefits. MS-1010, 5, 18, 370-371. Smith explains that a system VM provides a sandbox that isolates one system environment from others, that different system environments can provide isolation and enhanced security, and that major software systems running concurrently on the same hardware platform may be securely partitioned from one another. *Id.* Smith further explains that running applications in separate virtual machines improves robustness because erratic behavior, compromise, or failure in one guest system is less likely to affect software running on another guest system. *Id.* These teachings indicate why system-VM architectures were a well-known and practical approach for implementing multiple isolated operating-system environments on a shared hardware platform.

IX. OVERVIEW OF THE PRIOR ART COMBINATIONS

A. The Combination Of Khandekar And Bandhole

95. A POSITA implementing Khandekar's provisioning workflows would have sought techniques enabling remote, web-based interaction with the provisioning system from a variety of client devices. A POSITA also would have been interested in an administrable client/server framework that supports remote creation, operation, and management of allocated computing environments through a web interface and a supporting application-server tier. These interests follow from Khandekar's description that its provisioning workflows rely on a browser-accessed, network-facing user interface for initiating and managing provisioning tasks. Specifically, "[t]he user console is preferably a web server, with which the user interacts with any conventional web browser," and through this console, the user can select a "pre-configured, instantly deployable VM" or "custom-provision a VM" by submitting configuration selections to the provisioning server. MS-1004, 9:46-64, 13:66-14:3.

96. Khandekar also describes continued remote interaction after submission of a provisioning request: the task manager returns a task identifier, and the user can use the same browser-based console to check task status (and in some implementations, abort the task). MS-1004, 20:61-21:16. A POSITA would have understood these disclosures as describing a web-initiated, remotely operated

management workflow, which in turn implies a need for a concrete and scalable web-access architecture that works across common client devices. Put differently, once the control loop (submit → track/abort) is intended to run through a browser over a network, a POSITA would naturally look for standard web-tier patterns and backend integration approaches that make that remote workflow robust at scale. In that context, Bandhole's descriptions of remotely creating and operating a computing environment through "a web page on the Internet" provide compatible implementation teachings that fit directly into Khandekar's browser-based console approach. MS-1007, [0029], [0035], [0048], 16:35-39. Bandhole thereby supplies architectural detail for how browser requests can be handled and connected to server-side resource-management functions.

97. With that background, a POSITA working with Khandekar's VM provisioning system would have been motivated to incorporate Bandhole's web-based interaction and web-tier architecture as a straightforward way to strengthen the user-facing access path for provisioning and managing allocated environments at scale. Bandhole addresses the practical mechanics that a browser-driven console typically requires, such as receiving requests at a web server and relaying those requests to an application server that performs or coordinates the underlying management functions, implementation approaches that align with Khandekar's design premise of a network-facing, browser-accessed console. A POSITA

therefore would have treated Bandhole's remote UI and web-tier teachings as directly usable building blocks for implementing (and scaling) Khandekar's browser-based provisioning workflow.

B. The Combination Of Khandekar And Vasilevsky

98. The Khandekar-Vasilevsky System would have combined (i) Khandekar's VM provisioning workflow with (ii) Vasilevsky's approach for managing mappings between virtual storage constructs and underlying physical storage. Khandekar describes a provisioning environment where a provisioning server takes user requests through a console, selects an appropriate host using dynamically maintained host-state information, and then interacts with the chosen host through an API—for example, it can “query[] the host via an API” and “request the host (using the API) to reserve the amount of memory and disk space required” for the VM being provisioned. MS-1004, 12:57-13:8, 20:54-60. Vasilevsky, in turn, describes management functionality including a management agent (e.g., management agent 815) that maintains a “mapping database” defining mappings between virtual resources and physical hardware. MS-1005, [0150], Cl. 80. In the combined system, that storage mapping functionality would be integrated into the provisioning server environment so that, when storage capacity is reserved and allocated as part of provisioning, the system also records and maintains mapping information tying the storage-related virtual resources

associated with each provisioned instance to the physical storage resources used to satisfy those reservations. MS-1004, 20:54-21:24; MS-1005, [0150], Cl. 80.

99. In larger-scale or more dynamic virtualized deployments, a POSITA working from Khandekar would have recognized two practical needs. First, making the relationships between virtual resources and physical resources explicit (so they can be inspected and managed), and second, supporting flexible allocation and reallocation over time using common management interfaces (UI/CLI/API). Khandekar already emphasizes dynamic state tracking and API-driven resource actions during provisioning. It explains that “[i]nformation about the current state of the different hosts is preferably input dynamically into a database” used by the heuristics engine, and that the heuristics engine can “query[] the host via an API” and “[r]equest the host (using the API) to reserve the amount of memory and disk space required.” MS-1004, 12:57-13:8, 20:54-60. When a system is designed around continually changing host state and programmatic reservation of disk/memory, a POSITA would expect that the next engineering step is a clearer and more administrable representation of how virtual allocations correspond to physical capacity as conditions change during operation and across many provisioned assets. Teachings from Vasilevsky address that need, including by describing provisioning in real-time without restarting a virtual computing system. MS-1005, [0076].

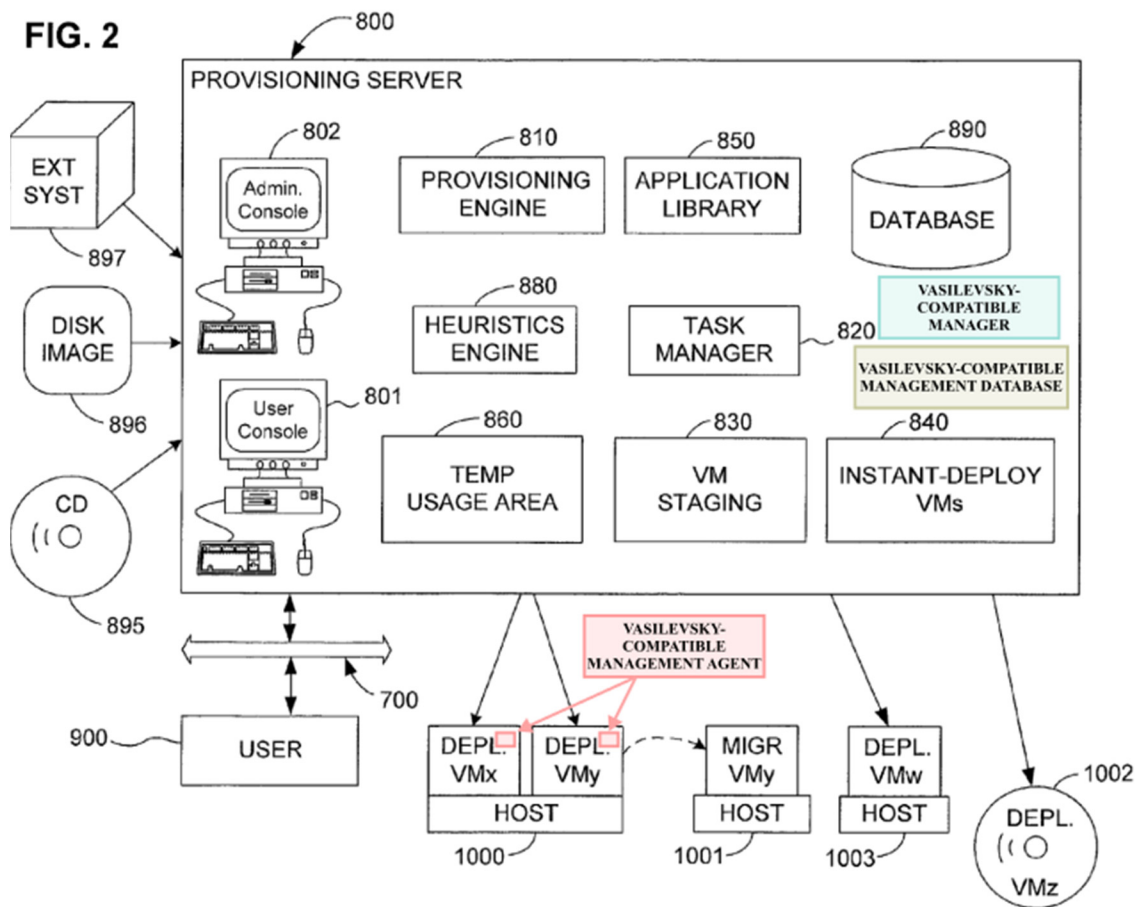
100. A POSITA implementing Khandekar's provisioning workflow would have been motivated to incorporate known virtualization-management techniques that make resource consumption easier to see and control, i.e., techniques that explicitly capture how requested virtual resources map onto and draw from underlying physical resources in a multi-host environment. MS-1004, 10:27-42, 21:26-31. That motivation follows naturally from Khandekar's own design: provisioning choices are driven by resource availability (e.g., the heuristics engine "determine[s] the [host] with enough memory and disk space," and "request[s] the host (using the API) to reserve the amount of memory and disk space required"), and the provisioning server treats provisioning as an orchestrated workflow whose progress can be monitored and controlled through its management plane. MS-1004, 20:54-21:24.

101. A POSITA therefore would have looked for established mechanisms that fit within the same management model Khandekar already uses (namely programmatic interfaces to hosts and management-console-driven control) and that provide a structured way to represent virtual-to-physical allocation relationships. Vasilevsky is technically pertinent in that respect because it describes a management agent that maintains a mapping database for virtual resources and physical hardware. MS-1005, [0084], [0150]. Incorporating that mapping approach into the Khandekar System would have been a compatible refinement with

predictable results and benefits for Khandekar's API-driven provisioning server: it supplies an established way to track and manage the resource-allocation relationships (including disk/memory reservations and assignments) that Khandekar already treats as core inputs and outputs of provisioning. MS-1004, 9:46-64, 20:54-60, 22:53-67.

102. In one example of the Khandekar-Vasilevsky combination, a provisioning server environment (e.g., Khandekar's provisioning server 800 and/or an associated management component) would have been configured to implement management functionality described in Vasilevsky (e.g., for management agent 815). This includes maintaining mapping information between storage-related virtual resources (e.g., a virtual storage entity) and the corresponding physical hardware (e.g., an actual storage device). MS-1004, 12:57-13:8, 20:54-60; MS-1005, [0150], Cl. 80. Operationally, when a user submits a provisioning request through the user console, Khandekar's heuristics engine uses dynamically maintained host-state information to select a host and can "query[] the host via an API" and "[r]equest the host (using the API) to reserve the amount of memory and disk space required" for the requested VM. MS-1004, 12:57-13:8, 20:54-60. As part of executing that reservation and allocation, the provisioning server environment would also generate or update mapping records for the provisioned instance so that the storage-related virtual resource allocated to that instance is

associated with the physical storage resource that actually provides the reserved disk capacity, consistent with Vasilevsky's disclosure that a management agent may maintain a "mapping database" defining mappings of virtual resources to physical hardware. MS-1005, [0150]; MS-1004, 20:54-60. The combined system would maintain a resulting storage mapping that records how the provisioned virtual storage construct corresponds to the actual storage device, allowing the system to track those virtual-to-physical storage relationships across provisioned assets. MS-1004, 20:54-21:24; MS-1005, [0150], Cl. 80.



MS-1004, FIG. 2 (modified to incorporate Vasilevsky's teachings)

103. Incorporating Vasilevsky's teachings would have provided a straightforward way for representing and controlling the relationship between virtualized constructs and underlying physical resources (including during execution) while keeping Khandekar's provisioning-server control plane and provisioning-task flow intact. A POSITA also would have reasonably expected the integration to be successful because it applies well-known virtualization-management techniques (explicit physical-to-virtual mapping and on-the-fly allocation through programmatic and/or UI-driven management) to enhance a provisioning workflow that already relies on defined host interfaces and an agreed protocol/API for provisioning requests and resource reservations.

C. The Combination Of Khandekar And Fujino

104. The Khandekar-Fujino System would have combined Khandekar's VM provisioning environment with Fujino's Storage Area Network (SAN) architecture in which storage network connectivity is represented in software (e.g., as SAN configuration information) by identifying the physical devices along a communication path (e.g., fiber channel switches and endpoint ports). More specifically, in the combined system, Khandekar's centralized provisioning server environment would be configured to maintain and use Fujino-style SAN configuration information as part of its inventory/state information for VM deployments that rely on shared storage accessible over a storage network. The

SAN configuration information would provide a software-defined representation of the storage communication paths between servers/hosts and shared storage by identifying the ports and intermediate fiber channel switches along each path, enabling the provisioning environment to define and track how provisioned VMs reach shared storage through the SAN.

105. A POSITA implementing Khandekar's provisioning workflow would have been motivated to incorporate Fujino's SAN connectivity management techniques to support implementations in which provisioned VMs rely on shared storage accessible over a storage network (e.g., a SAN). Khandekar contemplates such implementations through disclosure of a configuration in which "a shared storage area" is accessible by "a staging server as well as the host servers where the VMs are deployed." MS-1004, 16:64-17:13. Khandekar thereby suggests VM data may be maintained in shared storage accessible by multiple machines participating in provisioning and deployment. *See id.*, 16:64-17:13; *see* MS-1004, 9:26-32, 10:50-60. A POSITA would have therefore perceived Fujino's SAN architecture to be pertinent, since Fujino provides an implementation for network-accessible shared storage in which connectivity between endpoint devices is represented and maintained as configuration information. MS-1008, 5:31-45, 6:27-36, 10:16-30. A POSITA would have recognized these teachings as complementing Khandekar's VM deployments by supplying a way to define and

track the storage network connectivity used to reach shared storage from multiple computers/servers.

106. In the combined system, this SAN configuration information would be maintained as part of the provisioning server environment's inventory/state information (e.g., within, or in conjunction with, Khandekar's database 890), so that the system defines and tracks storage connectivity for deployments relying on a shared storage implemented via a storage network (e.g., SAN). MS-1004, 9:26-32, 10:50-60; MS-1008, 10:16-30. This would have enabled Khandekar's provisioned VMs (deployed on selected hosts) to utilize a software-defined representation of SAN connectivity that specifies the underlying physical ports and fiber channel switches implementing the storage communication path used for storage access. MS-1008, 6:27-36, 6:40-49, 10:16-30. In this manner, the combined system would provide a new mechanism defining and tracking storage within the provisioning server's inventory for use in coordinating VM deployments. MS-1004, 18:2-5, 20:54-60.

107. A POSITA would have understood that when VM deployments rely on shared storage accessible from multiple servers, provisioning benefits from maintaining a software representation of the storage-network connectivity used to reach that shared storage. This allows connectivity to be defined, tracked, and referenced as part of provisioning and deployment coordination. MS-1008, 10:16-

30. This would have also motivated a POSITA to incorporate Fujino's SAN configuration information (identifying the physical ports, switches, and path elements) into inventory/state information maintained by Khandekar's provisioning environment to support deployments using SAN-accessible shared storage.

108. A POSITA would have reasonably expected to succeed in combining the teachings of Khandekar and Fujino because the combination incorporates Fujino's SAN connectivity representation (as configuration information identifying physical ports and switches) as inventory/state information used for VM provisioning, yielding predictable results in how storage network connectivity is specified and maintained.

D. The Combination of Hirschfeld And Smith

109. The Hirschfeld-Smith System would have predictably combined Hirschfeld's virtualization environment with Smith's well-known teachings regarding implementation of virtualization software using a VMM to manage multiple virtual machines on a host platform.

110. Hirschfeld describes configurations in which a physical server supports multiple logical servers. For example, Hirschfeld explains that "logical servers 106 form[] a logical server cloud 201," and more specifically that "the logical servers LS 1-4 are linked to the [physical server (PS) A], the logical servers

LS 5-8 are linked to the PS B, and the logical servers LS 9-N are linked to the PS C.” MS-1006, 7:63-8:10. Hirschfeld further describes that each logical server executes its own operating system, which may be “any desired OS,” such as operating systems “from Microsoft or based on UNIX, Linux, etc.” Id., 10:9-31. In the combined system, Hirschfeld’s logical-server-cloud architecture would be implemented using Smith’s VMM-based virtualization approach so that the multiple logical server operating systems contemplated by Hirschfeld are managed through a software layer that controls allocation of and access to the host platform’s hardware resources. MS-1010, 4, 11, 369.

111. A POSITA working from Hirschfeld would have recognized that Hirschfeld describes the high-level virtualization architecture and the existence of multiple logical servers on shared physical hardware, while leaving the precise software mechanism for managing those multiple logical servers on a single physical server as an implementation detail. MS-1006, 9:47-57.

112. A POSITA would have therefore looked to well-known virtualization mechanisms used in similar environments as of the Critical Date to implement that management functionality. Smith is technically pertinent in that respect because Smith describes a “system VM environment” in which “real resources of the host platform are shared among the guest system VMs, with a layer of software, the virtual machine monitor (VMM), managing the allocation of, and access to, the

hardware resources of the host platform.” MS-1010, 369. Smith also explains that “[a] virtual machine (VM) is implemented by adding a layer of software to a real machine to support the desired virtual machine’s architecture,” and that such “virtualizing software is often referred to as the virtual machine monitor (VMM).” Id., 4, 11. A POSITA would have understood Smith’s VMM teachings as providing a known and compatible mechanism for implementing the kind of shared-host, multiple-virtual-instance environment Hirschfeld contemplates.

113. A POSITA would have been motivated to incorporate Smith’s VMM teachings into Hirschfeld because Hirschfeld’s architecture presupposes multiple independent logical server operating systems running on the same physical server hardware. MS-1006, 7:63-8:10, 10:9-31. When a system is designed so that multiple logical servers share one physical server while each maintains its own operating system environment, there is a practical need for a software layer that mediates use of the host’s processor, memory, storage, and other physical resources among those separate virtualized instances. Smith addresses exactly that need by describing a VMM that manages allocation of, and access to, host hardware resources among multiple guest operating systems on the same hardware platform. MS-1010, 369. A POSITA would have therefore understood Smith’s VMM disclosures as supplying an implementation mechanism for Hirschfeld’s contemplated multi-logical-server configurations rather than changing Hirschfeld’s

basic architecture.

114. In one example of the Hirschfeld-Smith combination, Hirschfeld's virtualized logical server cloud would be implemented with its SCM-based control framework and bank of physical servers as described by Hirschfeld, while each physical server would execute a Smith-like VMM to support and manage multiple logical operating systems corresponding to Hirschfeld's logical servers. MS-1006, Fig. 1, 7:63-8:10, 8:39-64; MS-1010, 369. In that combined system, the Smith-like VMM would provide functionality of the type Smith describes, including "manag[ing] the allocation of, and access to, the hardware resources of the host platform." MS-1010, 369. The VMM would also make the real system resources available to one or more guest operating systems executing on the same hardware such that each guest operating system is given the appearance of owning the resources it allocates to its user programs. *Id.* This is technically consistent with Hirschfeld's disclosure that multiple logical servers operate on shared physical-server resources while maintaining logical separation and independent operating-system environments. MS-1006, 2:1-7, 10:9-31.

115. A POSITA would have reasonably expected success in combining Hirschfeld and Smith in this manner. The combination applies a well-known software element of virtualization systems, namely Smith's VMM, within a well-known virtualization environment, namely Hirschfeld's SCM-controlled logical-

server cloud operating across one or more physical servers. MS-1006, 7:63-8:10, 10:9-31; MS-1010, 4, 11, 369. Smith's teachings do not require reworking Hirschfeld's logical-server-cloud concept. Instead, they provide a known way to implement the host-side virtualization management needed for Hirschfeld's multiple-logical-server-per-host configurations. A POSITA would have understood this combination as a predictable use of known virtualization technology to implement a known virtualization architecture, with predictable results in supporting configurations where a physical server hosts multiple logical servers under centralized management.

E. The Combination of Hirschfeld, Smith, And Bandhole

116. The Hirschfeld-Smith-Bandhole System would have predictably combined Hirschfeld's virtualization environment with well-known client device types for accessing remotely managed virtualized resources. Hirschfeld describes a virtualization environment in which users access logical servers under control of an SCM through remote "terminals," and explains that those terminals may be implemented as "simple terminals," "thin client computers," and "low end PCs." MS-1006, 6:28-63. Bandhole, in turn, confirms that client-side access of this kind was well-known by the Critical Date and describes computing devices that "can be made in many dimensions and styles," including "workstation," "laptop," and "palmtop" form factors. MS-1007, [0035], [0040]. In the combined system,

Hirschfeld's remote-terminal access model would therefore be implemented using known client-device form factors of the type Bandhole describes, while preserving Hirschfeld's SCM-based management and logical-server access architecture.

117. A POSITA implementing Hirschfeld's terminal-based user access to the SCM and logical servers would have been motivated to enable use of the terminals from these known client device types because Hirschfeld's approach presupposes remote client communication over a network and expressly contemplates modest client endpoints rather than specialized hardware. MS-1006, 6:28-63. Hirschfeld's disclosure of "simple terminals," "thin client computers," and "low end PCs" indicates that the client side of the system is intended to be flexible and lightweight, with the principal management and virtualization functionality residing in the server-side environment. *Id.* A POSITA would have understood from that architecture that the particular client-device form factor was not critical so long as the device could support the remote communication needed to reach the SCM and user interface. Bandhole's teachings are technically pertinent in that respect because they provide concrete examples of well-known endpoint devices suitable for remote interaction with server-side computing environments, including workstation, laptop, and palmtop/PDA form factors. MS-1007, [0029], [0035], [0040], [0048].

118. A POSITA would have further recognized that Bandhole's device-

form-factor teachings are complementary to Hirschfeld's terminal model rather than changing it. Hirschfeld already contemplates networked client devices of varying capability levels for reaching the SCM and logical servers. MS-1006, 6:28-63. Bandhole simply confirms that such client access could be carried out from a range of known computing-device form factors used for remote interaction with managed computing environments. MS-1007, [0035], [0048]. A POSITA would have understood that implementing Hirschfeld's terminal-based SCM access on workstation, laptop, or palmtop/PDA devices would not require changes to Hirschfeld's underlying virtualization and control sequence for managing logical servers on physical resources, because those teachings affect the type of client endpoint used to reach the management interface, not the server-side mechanisms by which the SCM controls the logical server cloud. MS-1006, 6:28-7:30.

119. In one example of the Hirschfeld-Smith-Bandhole combination, a user would access Hirschfeld's SCM and logical servers from a client device implemented as a workstation, laptop, or palmtop/PDA, each communicating over the network with the SCM's user interface to log in and obtain access to logical servers within the cloud. MS-1006, 6:28-63; MS-1007, [0035], [0048]. In that combined system, the client device form factor would be a predictable implementation choice for Hirschfeld's disclosed terminal while the server-side logical-server-cloud and SCM-control architecture would remain as Hirschfeld

describes it. This would allow the same remote management and access model to be used from a broader set of conventional client devices without altering the logical-server virtualization environment itself.

120. A POSITA would have reasonably expected success in combining Hirschfeld and Bandhole in this manner. The combination applies well-known client device form factors to a known remote-access virtualization environment that already contemplates lightweight, network-connected client endpoints. MS-1006, 6:28-63; MS-1007, [0035], [0048]. Because the combination concerns the predictable use of known endpoint-device implementations for accessing an existing SCM-based remote management interface, a POSITA would have understood it as a straightforward application of known client-side access technology to Hirschfeld's existing virtualization framework, with predictable results in enabling access from workstation, laptop, and palmtop/PDA devices.

F. The Combination of Hirschfeld, Smith, And Vasilevsky

121. The Hirschfeld-Smith-Vasilevsky System would have predictably combined Hirschfeld's virtualized logical-server environment with Vasilevsky's well-known techniques for mapping and dynamically allocating virtual resources, including storage resources, to underlying physical hardware. Hirschfeld describes a virtualized environment in which storage resources "may be divided into multiple physical and logical devices (e.g., File Transfer Protocol (FTP) and Network

Attached Storage (NAS) or the like),” and further explains that “[a]ccess to the network and storage resources is preferably logical rather than physical so that they act like resource clouds....” MS-1006, 8:66-9:20. Those teachings indicate that Hirschfeld already contemplates a virtualization architecture in which storage is presented and used as a logical resource rather than as a directly exposed physical device. In the combined system, Vasilevsky’s resource-mapping and management techniques would be incorporated into Hirschfeld’s SCM-controlled environment so that logically accessed storage resources associated with logical servers are managed through explicit virtual-to-physical mapping and dynamic resource-allocation mechanisms.

122. A POSITA would have been motivated to combine these teachings because Hirschfeld’s storage-resource-cloud model naturally gives rise to a practical need for management mechanisms that make storage assignment more visible, controllable, and administrable within the virtualized environment. Hirschfeld’s disclosure that storage resources are divided into logical and physical devices and accessed logically rather than physically would have suggested to a POSITA that the environment benefits from a structured way to define how those logical storage resources correspond to the underlying physical storage hardware. MS-1006, 8:66-9:20. That would have prompted a POSITA to look for well-known virtualization-management techniques for implementing logical storage

capabilities, including configurations in which a logical server provides NAS-like storage services backed by underlying physical storage. Vasilevsky is technically pertinent in that respect because it describes management of virtualized resources through management interfaces and mapping structures that associate virtual resources with actual physical hardware. MS-1005, Abstract, [0009], [0014], [0074]-[0079], [0150].

123. A POSITA implementing Hirschfeld's SCM-based control of logical servers would have further recognized that Vasilevsky's storage-related teachings address the same general technical problem presented by Hirschfeld's logical-storage model. Hirschfeld describes logical access to storage resources within a managed virtualized environment, while Vasilevsky describes management frameworks that improve transparency and controllability of virtual-resource assignment by using management-side interfaces, allocation controls, and mappings between virtual resources and physical hardware. MS-1006, 8:66-9:20; MS-1005, [0077]-[0083], [0150], [0166]. A POSITA would have understood that incorporating Vasilevsky's teachings into Hirschfeld's environment would provide a more explicit and administrable way to control how virtualized storage resources are assigned to, and used by, logical servers. That is, Vasilevsky would supply a known mechanism for making Hirschfeld's logically accessed storage resources easier to configure, allocate, and track within the SCM-controlled virtualization

environment.

124. In one example of the Hirschfeld-Smith-Vasilevsky combination, Hirschfeld's logical server cloud would continue to operate with logical servers under SCM control, while Vasilevsky-like management functionality would be used to maintain and control mappings between storage-related virtual resources associated with those logical servers and the underlying physical storage hardware that supports them. MS-1006, 8:66-9:20; MS-1005, [0074]-[0079], [0150]. In that combined system, when storage is allocated to a logical server or when a logical server is configured to provide storage-oriented services, the management framework would maintain a representation of the virtual storage resource and its association with the underlying actual storage device or devices. This is technically compatible with Hirschfeld's disclosure because Hirschfeld already describes storage as a logically accessed cloud-like resource, and Vasilevsky provides a known way to implement and manage that kind of logical storage abstraction through management interfaces and virtual-to-physical mapping structures.

125. A POSITA would have reasonably expected success in combining Hirschfeld and Vasilevsky in this manner. Vasilevsky's storage-related teachings are complementary to Hirschfeld's SCM-controlled virtualization paradigm and do not require altering Hirschfeld's basic concept of running isolated logical servers on shared physical infrastructure. MS-1006, 8:66-9:20; MS-1005, [0074]-[0079],

[0150]. Instead, the combination applies well-known virtualization-management mechanisms, including mapping databases, management interfaces, and dynamic allocation or deallocation of resources, to a virtualization environment that already contemplates logical, cloud-like access to storage resources. A POSITA would therefore have understood the combination as a predictable use of known virtualization-management techniques to implement and manage virtual storage capabilities within Hirschfeld's existing logical-server-cloud architecture.

G. The Combination of Hirschfeld, Smith, And Fujino

126. The Hirschfeld-Smith-Fujino System would have predictably combined Hirschfeld's virtualization environment with Fujino's well-known Storage Area Network (SAN) configuration and management techniques for logically representing network connectivity. Hirschfeld describes a virtualized environment in which the shared network "may be segmented into multiple network fabrics" and further explains that, when logical servers are moved, "network configuration changes may be required to maintain connectivity," including that "network hardware mappings (e.g., port mappings) are modified as necessary to maintain the logical net configuration." MS-1006, 8:39-64, 15:10-17. Those teachings indicate that Hirschfeld already contemplates a virtualization environment in which logical network connectivity must be maintained across reconfiguration events and in which underlying network-hardware relationships

matter to preserving the logical network view presented to the logical servers. In the combined system, Fujino's SAN configuration techniques would be used to represent and manage those network-connectivity relationships in maintained software information that identifies the physical devices along the relevant communication paths.

127. A POSITA would have been motivated to combine these teachings because Hirschfeld's disclosure of segmented network fabrics and modified port mappings naturally gives rise to a practical need for a more structured way to define, track, and preserve network connectivity as the system changes. MS-1006, 8:39-64, 15:10-17. When a virtualization environment depends on maintaining a "logical net configuration" while logical servers are moved or otherwise reconfigured, a POSITA would have looked for known techniques that represent the relevant connectivity relationships in a form that can be managed and updated through the control architecture rather than inferred each time from the underlying hardware layout. Fujino is technically pertinent in that respect because Fujino describes SAN management using maintained configuration information and related management information identifying the physical network devices, such as Fibre Channel switches and endpoint ports, along communication paths between host devices and storage control devices. MS-1008, 5:31-45, 6:27-36, 6:40-49, 7:35-43.

128. A POSITA implementing Hirschfeld's SCM-based control of logical servers would have further recognized that Fujino's SAN configuration teachings address the same general technical problem implicated by Hirschfeld's disclosure of maintaining logical network connectivity through hardware mappings.

Hirschfeld describes a managed virtualized environment in which network hardware mappings may be modified as necessary to preserve logical connectivity during reconfiguration. MS-1006, 15:10-17. Fujino, in turn, describes configuration information that identifies the ports, switches, and communication-path elements that implement storage-network connectivity and uses that information as part of SAN management. MS-1008, 7:24-34, 15:21-35, 27:27-41.

A POSITA would have understood that incorporating Fujino's teachings into Hirschfeld's environment would improve predictability and controllability of network connectivity for storage access and related communications because the environment would have a maintained software representation of how the relevant physical network devices implement the communication paths needed by the system.

129. In one example of the Hirschfeld-Smith-Fujino combination, Hirschfeld's logical server cloud would continue to operate under SCM control, while Fujino-style SAN configuration information would be maintained to represent the communication paths used for storage-network connectivity between

physical servers hosting logical servers and storage-side devices. MS-1006, 15:10-17; MS-1008, 6:27-36, 7:35-43, 27:27-41. In that combined system, the relevant port topology and communication-path information would be represented in maintained configuration information that identifies the physical ports and switches implementing the path. This would provide the SCM-controlled virtualization environment with a structured way to preserve, update, and reason about network connectivity as logical servers are moved or otherwise reconfigured. That is technically compatible with Hirschfeld because Hirschfeld already contemplates maintaining logical network configuration through changes to network hardware mappings, and Fujino provides a known way to represent and manage those hardware-level connectivity relationships in software.

130. A POSITA would have reasonably expected success in combining Hirschfeld and Fujino in this manner. Fujino's SAN configuration teachings complement Hirschfeld's SCM-controlled virtualization paradigm and do not require changing Hirschfeld's fundamental concept of running isolated logical servers on shared physical infrastructure. MS-1006, 15:10-17; MS-1008, 6:27-36, 7:35-43, 27:27-41. Instead, the combination applies well-known SAN management mechanisms, including maintained configuration information describing port topology and communication paths, to a virtualized and networked environment that already contemplates preserving logical network connectivity through port-

mapping changes during reconfiguration. A POSITA would therefore have understood the combination as a predictable use of known SAN-management techniques to implement and manage the network-connectivity relationships already implicated by Hirschfeld's logical network configuration model.

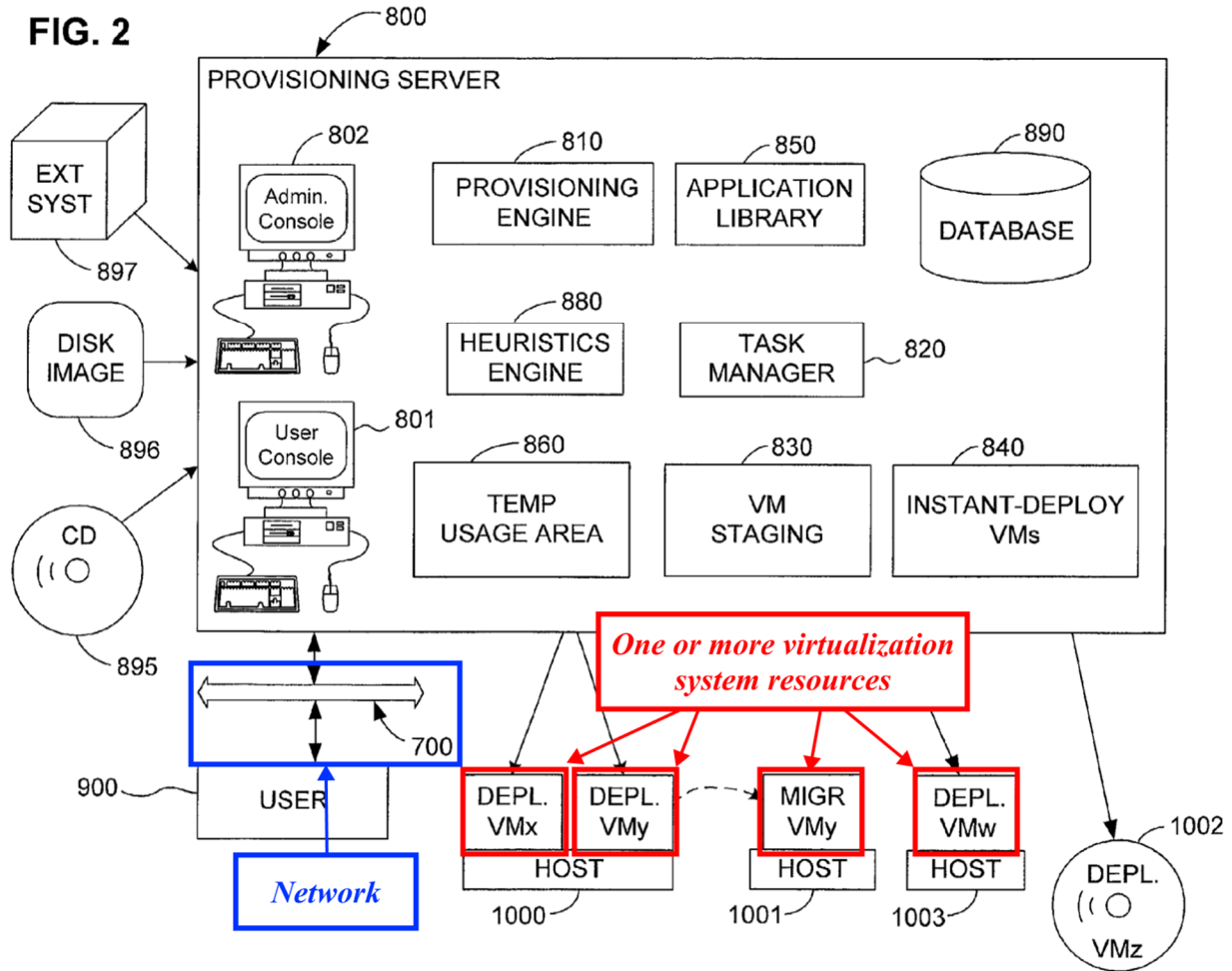
X. THE PRIOR ART MAKES THE CHALLENGED CLAIMS OBVIOUS

A. Grounds 1A/1B: Claims 1 And 2 Are Anticipated And/Or Rendered Obvious By Khandekar

1. Claim 1

[1.pre] A method for access to one or more virtualization system resources over a network, the method comprising:

131. The Khandekar System satisfies the requirements of [1.pre]. In Khandekar, a user interacts with a browser-based user console to request and access virtual machines deployed on host servers within a virtualized environment. MS-1004, 9:46-64, 13:66-14:3, 14:4-8. The Khandekar system enables access to one or more VM resources (e.g., resources of VMx, VMy) (***one or more virtualization system resources***) over a network (e.g., 700) (***network***).



MS-1004, FIG. 2¹

[1.1] *implementing a virtualization system that has a control computer and one or more physical resources and virtual assets connected to the control computer,*

132. The Khandekar System (*virtualized system*) includes a provisioning server 800 (*control computer*) that communicates over a network with one or more host computers/servers (e.g., 1000, 1001) that supply the underlying hardware

¹ Annotations to figures throughout this petition are shown in colored markings and text.

compute resources (*one or more physical resources*) and execute VMs deployed by the provisioning server (*virtual assets*). MS-1004, 6:1-63, 7:7-27, 9:26-64, 10:27-42, 13:17-35, 17:25-39, 19:33-45, 21:50-52, 21:26-52.

133. As I discussed for [1.pre], Khandekar explains that the “VM provisioning server 800” manages a plurality of “Host” computers—i.e., the hardware machines on which virtual machines are deployed. MS-1004, 9:26-32, 10:27-42. Each host runs a virtual machine monitor (VMM), and the VMM “virtualizes all, or at least some of, the resources of the machine.” *Id.*, 7:7-27, 13:17-35. Khandekar further describes the VMM as “usually a thin piece of software that runs directly on top of the host,” i.e., executed on the physical host hardware. *Id.* In that arrangement, the host servers provide the underlying compute infrastructure and correspond to the recited physical resources, while the deployed “virtual machines (VMs)” instantiated on those hosts under the direction of provisioning server 800 correspond to the recited virtual assets.

134. Khandekar also describes that provisioning server 800 communicates with the host machines over a network (e.g., 700) and deploys VMs to selected hosts in response to user requests. MS-1004, 6:1-12, 9:46-64, 10:27-42. For instance, during provisioning, the provisioning engine “deploys a VM on a host” and invokes a host API to register the new VM, which associates the VM instance with the selected physical host. *Id.*, 19:33-45, 21:50-52, 21:26-52. Because

provisioning server 800 initiates and coordinates VM deployment, configuration, and lifecycle actions on the hosts, a POSITA would have understood the VMs to be connected to, and subject to control by, the provisioning server. The Khandekar System reflects a virtualization system in which a centralized control computer manages and communicates with multiple physical servers executing virtual assets.

135. A POSITA would have implemented the Khandekar System using Khandekar's dedicated network server hardware. MS-1004, 9:39-45. A POSITA would have avoided deploying the control computer as a virtual asset so that management functions remain available as dedicated, out-of-band resources and do not compete with tenant VMs for host compute cycles.

[1.2] wherein each of the one or more physical resources execute at least two virtual assets that comprise software instantiations of computer systems, and

136. The Khandekar System includes host servers (e.g., Khandekar's host servers 1000, 1001) (***physical resources***) that each execute at least two VMs (***at least two virtual assets***) that include software that can virtually perform functions of physical devices (***software instantiations of computer systems***). MS-1004, 6:24-61, 7:44-47.

137. As discussed for [1.1], Khandekar explains that a "virtual machine (VM)" is a "software abstraction" of a "physical computer system." MS-1004, 6:24-42. Consistent with that description, Khandekar explains that a VM can

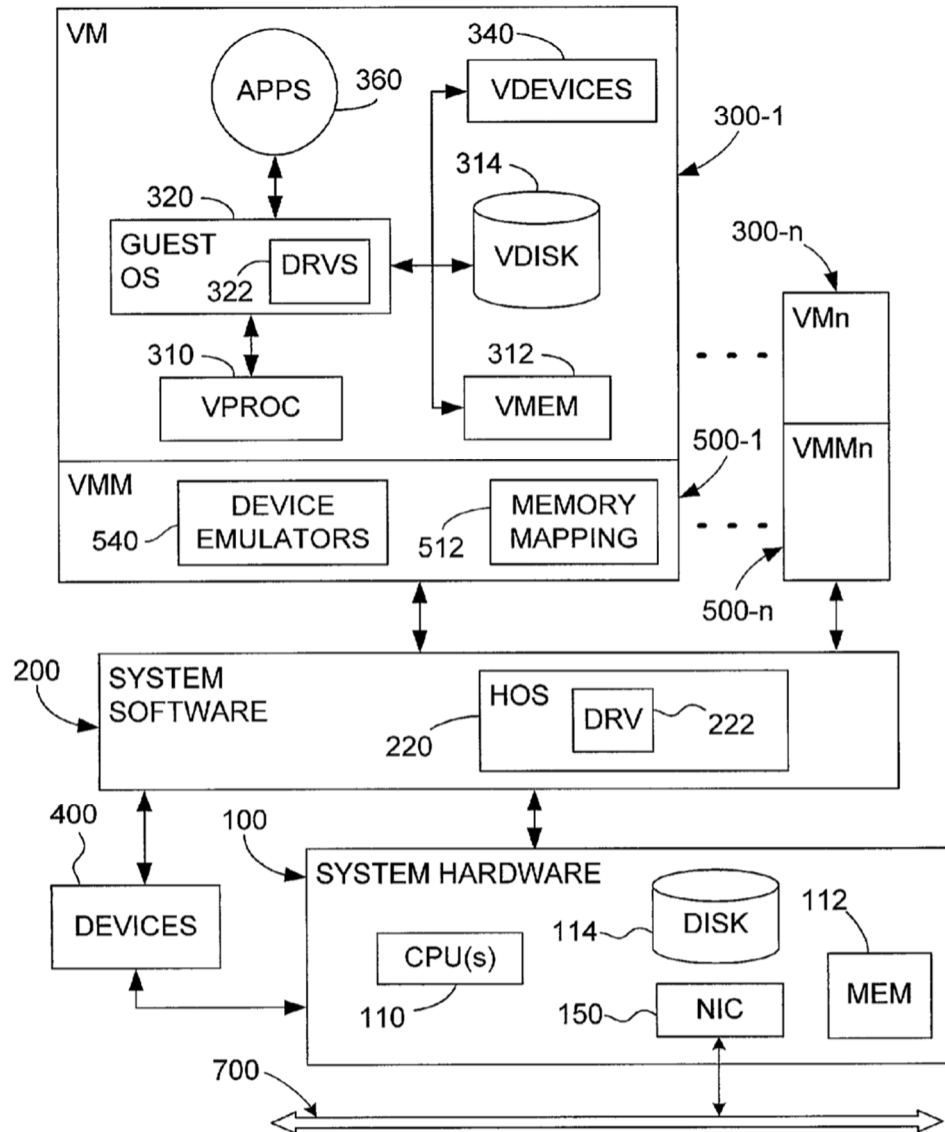
include software constructs that emulate physical hardware, such as “optional virtual peripheral devices 340 (VDEVICES)” that are “implemented in software to emulate the corresponding components of an actual computer.” *Id.* As another example, the Khandekar System includes “device emulators 540 that emulate the characteristics of respective physical devices.” *Id.*, 7:48-55. In technical terms, these disclosures describe VM software modules that model and reproduce device behavior in software, such that the VMs in the Khandekar System include software that can virtually perform functions of physical devices (*software instantiations of computer systems*).

138. A POSITA also would have found it straightforward to configure the Khandekar System so that each host server executes at least two VMs. For example, if host server 1000 is the “one or more physical resources,” then its execution of VMx and VMy (“*at least two virtual assets that comprise software instantiations of computer systems*”). In another example, when identifying a prior art configuration in Figure 1, Khandekar discloses “a plurality of virtual machines (VMs)” 300-1 to 300-n that run on the system software 200, thereby revealing that configurations in which multiple VMs are run on a single host server were known as of the Critical Date. MS-1004, 5:52-57.

139. A POSITA would have also understood that running multiple VMs on a single host server was not an unusual or specialized edge case, but one of the

basic design purposes of virtualization systems. Khandekar describes VMs as software abstractions of physical computer systems running on shared host resources. The point of that architecture is to permit the host's processor, memory, storage, and other resources to be shared among multiple independently operating VM instances rather than dedicating the full machine to only one software environment. MS-1004, 5:52-57, 6:24-42, 7:7-27. In that context, a POSITA would have understood that there would be many ordinary deployment scenarios in which a host server would, as a matter of course, execute at least two VMs, because that is how virtualization systems achieve consolidation, flexibility, and more efficient use of physical hardware.

FIG. 1
(Prior Art)



MS-1004, FIG. 1

[1.3] *the virtual assets are managed by virtualization layer instances running on each of the physical resources;*

140. The Khandekar System would have provided host servers (e.g., 1000,

1001) (*physical resources*) that execute respective VMMs (*virtualization layer instance*) that manage the virtual machines or logical servers (*virtual assets*) executing on that host. MS-1004, 6:63-7:6, 7:7-27, 13:17-35.

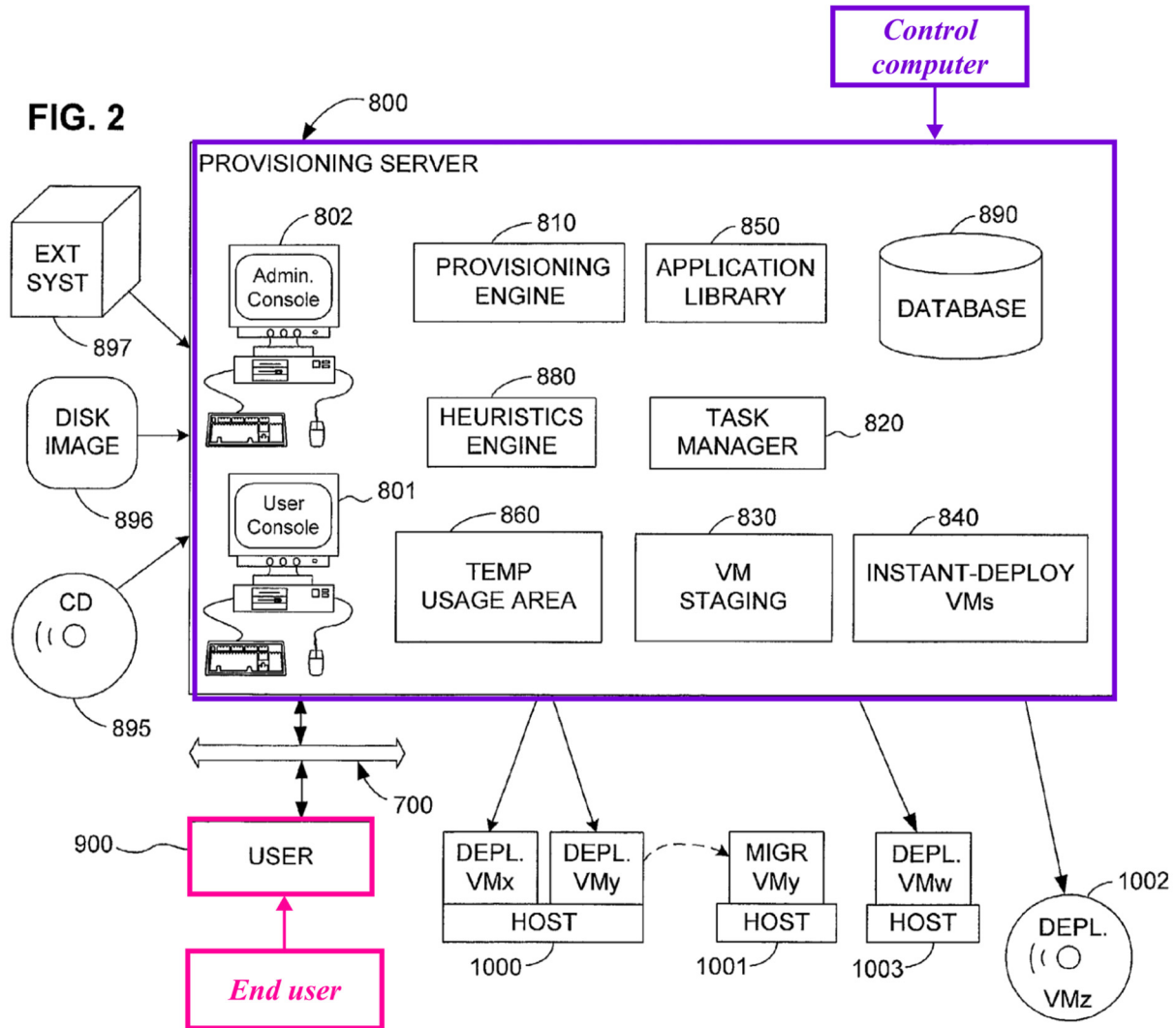
141. As I discussed for [1.1] and [1.2], Khandekar describes that a “virtual machine monitor (*VMM*)” is “pre-installed on each host” within a “plurality of hardware hosts.” MS-1004, 6:63-7:6, 13:17-35. The VMM “virtualizes all, or at least some of, the resources of the machine,” and is “usually a thin piece of software that runs directly on top of the host.” *Id.*, 7:7-27. In this architecture, the VMM running on a given host represents a software layer that manages the virtual machines executing on that host, including supporting VM execution and VM lifecycle operations on that physical resource. A POSITA would have understood that supporting VM execution and related VM operations on the host involves management of those VMs, because the VMM enables the VM to run, coordinates its access to host resources, and maintains the virtualized operating environment on which the VM depends. In this architecture, the VMM running on a given host therefore represents a software layer that manages the virtual machines executing on that host, including by supporting VM execution and other VM operations on that physical resource.

[1.4] connecting an end user to the control computer; and

142. The Khandekar System connects a user (e.g., *end user*) (via a web

browser) to the provisioning server 800 (*control computer*) through a user console 801. That console serves as the network-accessible interface through which the user submits requests to, and otherwise interacts with, the provisioning server, thereby providing a networked connection between the user and the provisioning server. MS-1004, 1:44-63, 9:46-64, 13:66-14:3.

143. As I discussed for [1.pre], Khandekar describes that an end user uses the user console to interact with the provisioning system, and that the user console is “preferably a web server, with which the user interacts with any conventional web browser.” MS-1004, 13:66-14:3. Through that browser-based interface, the user “logs in to the provisioning server” and then either selects a pre-configured virtual machine or provides configuration selections for a custom-provisioned VM. *Id.* This browser-based interaction establishes a network connection between the user and the provisioning server established through the user console, which connects the user to the provisioning server for submitting requests and receiving responses. *Id.*, 1:44-63, 9:46-64.



MS-1004, FIG. 2

[1.5] *accessing, through the control computer connection, the virtual assets by the end user.*

144. The Khandekar System enables the end user, through the user's connection to the provisioning server 800 (*control computer*) via the user console 801, to access the VMs (*virtual assets*) deployed on and executed by the host servers (*physical resources*). MS-1004, 9:65-10:14, 13:66-14:18, 15:29-39, 20:64-

21:8, 21:9-15.

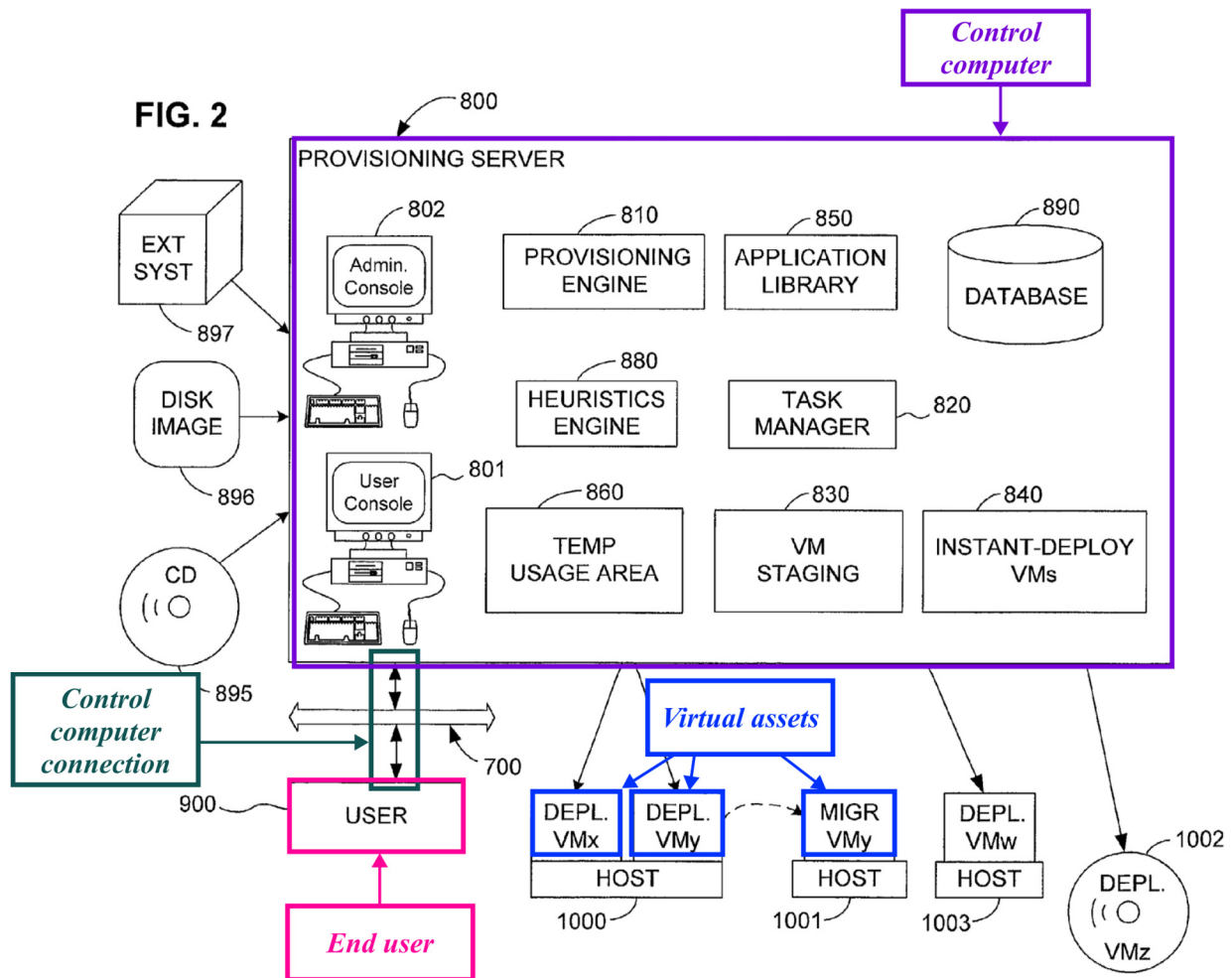
145. As I discussed for [1.4], Khandekar describes that the user connects to the provisioning server through the user console, which is “preferably a web server...” MS-1004, 13:66-14:3. In the Khandekar System, the user console supports access to VMs through a connection provided by the provisioning server 800 (e.g., by logging into server directly). Put another way, the provisioning server 800 can provide the network-facing access point used for VM access, such that communications to reach the VMs are handled through the provisioning server as an intermediary or gateway in the data path, and the user’s access to the VMs is routed directly through the provisioning server.

146. For example, through this connection, the user can may access VM-related resources by selecting a pre-configured VM (e.g., an “instantly deployable” VM) or submitting configuration selections (e.g., OS configurations such as hostname, IP address, and virtual hardware parameters) for a custom-provisioned VM. *Id.*; *see id.*, 9:65-10:14, 14:4-18.

147. Khandekar describes continued use of the same console/browser connection after a provisioning request is submitted: the system returns a task identifier, and the user can use the console to query the task status and, in some cases, abort the task before completion. *Id.*, 15:29-39, 20:64-21:8, 21:9-15.

Khandekar also explains that, once the VM is created, the user is informed (e.g.,

“via the browser from which he submitted the request”). *Id.*, 15:29-39. After deployment, Khandekar notes that “[t]he user can then access the newly created VM via any known remote access software,” e.g., “...software available from VMware.” *Id.*, 14:4-18. Accordingly, the user’s connection to the provisioning server is an example of a control computer connection that enables the user to access the provisioned VM (*virtual asset*).



MS-1004, FIG. 2

2. *Claim 2*

[2] The method of claim 1, wherein connecting the end user to the control computer further comprises connecting the end user to the control computer over one of a network, a wide area network or the Internet.

148. The Khandekar System connects a user 900 (*end user*) to the provisioning server 800 (*control computer*) through a network 700, including implementations where the network is the Internet. MS-1004, 9:46-64. Khandekar describes that “[o]ne or more users 900 communicate with the provisioning server 800 via...a network 700, such as the Internet, or over any other known channel.” MS-1004, 9:46-64.

B. Ground 1C: Claims 3-5 Are Obvious Over The Combination of Khandekar And Bandhole

1. *Claim 3*

[3] The method of claim 2, wherein connecting the end user to the control computer further comprises providing the end user, a desktop, a laptop or a personal digital assistant (PDA) and client resources.

149. The Khandekar-Bandhole System would have permitted a user to leverage a connection to a provisioning server (e.g., 800) (*connecting the end user to the control computer*) through a “conventional web browser,” which a POSITA would have understood to be accessible from known device types (e.g., “workstation” (*desktop, client*), “laptop” (laptop), “palmtop” (*personal digital assistant (PDA)*)). MS-1004, 9:46-64, 13:66-14:3; MS-1007, [0035].

150. Khandekar describes that its user console 801 can be accessed using

“any conventional web browser.” MS-1004, 13:66-14:3. Although Khandekar does not list particular device categories, to a POSITA the technical implication of using a browser to reach a remote, network-facing console is that the client device can be any standard platform capable of running such a browser. Bandhole makes that point explicit in describing a client/server system where a client computer communicates over a network (e.g., the Internet) with server-side components (e.g., a web server and/or application server) to access remotely allocable resources and services. MS-1007, [0004], [0047], [0048]. Bandhole further explains that computing devices can come in many “dimensions and styles,” including “workstation,” “laptop,” and “palmtop” form factors. *Id.*, [0035]. In the Khandekar-Bandhole System, the end user's browser-based access to a user console (e.g., 801) would therefore have been implemented using a client device of a known form factor, such as a “workstation” (*desktop, client*), “laptop” (*laptop*), or “palmtop” (*personal digital assistant (PDA)*).

2. Claim 4

[4] The method of claim 3, wherein accessing the virtual assets further comprising accessing an application server by the end user.

151. The Khandekar-Bandhole System would have enabled a user to access a VM (*virtual asset*) through a connection to a provisioning server (e.g., 800), and a POSITA would have understood that the accessed VM can provide application server functionality when they are configured with application software. MS-1004,

9:46-64, 14:4-18, 23:11-20. As I discussed for [1.5], Khandekar gives concrete examples of user access through that connection, such as supplying configuration selections for a VM and checking (via the console) the status of a provisioning task. MS-1004, 9:65-10:14, 11:1-10, 15:29-39, 20:64-21:8.

152. As I discussed for [1.5], Khandekar describes that a user connects to provisioning server 800 through user console 801 using, for example, a web browser, and through this connection, the user may access VM-related resources (e.g., select a pre-configured VM, submit configuration selections, query provisioning task status) and is informed upon completion. MS-1004, 9:46-64, 11:1-10, 13:66-14:3, 15:29-39. After deployment, Khandekar explains that “the user can then access the newly created VM via any known remote access software, such as the remote console software...” *Id.*, 14:4-18. From an implementation perspective, the console-to-provisioning-server connection is the management-plane path that the user uses to request provisioning and obtain access information (or initiate access), and the user then reaches the deployed virtual asset using the access mechanism the system makes available. *Id.*, 13:66-14:18.

153. Khandekar also describes a VM representing an application server when describing a web application that uses “an application server on another [computer].” MS-1004, 23:11-20. In the Khandekar-Bandhole System, the server-cloud architecture would have enabled logical server instances to be accessed by

users through, for instance, Khandekar's user portal, and such accessed instances may include application-server functionality by virtue of being configured with application software. *Id.*

3. *Claim 5*

[5]² The method of claim 1, wherein the virtual asset is providing end user access to one of an application server, to a middleware server and to a database.

154. The Khandekar-Bandhole System would have enabled a user to access a VM (*virtual asset*) through a connection to the provisioning server and, in some implementations, that VM provides end user access to an *application server* or to a *database*. MS-1004, 11:50-58, 23:11-20. Khandekar describes a web application that uses “an application server on another [computer],” which is consistent with a VM configured to provide or support *application server* functionality. *Id.*, 23:11-20. Khandekar also describes that a VM may run an application representing a “database server,” thereby providing end user access to a *database* when the VM runs that server application. *Id.*, 11:50-58.

155. As discussed for [4], Khandekar describes a provisioning environment in which a user accesses a provisioned VM through the provisioning workflow and

² Because of the phrase “one of,” satisfaction of claim 5 does not mandate providing end user access to each of the enumerated elements recited in the claim.

then accesses the resulting VM through remote access software. MS-1004, 14:4-18, 23:11-20. Khandekar also describes that users may select application software as part of the VM provisioning process. *Id.*, 9:65-10:14, 11:50-58. A POSITA would have understood from that architecture that the functionality made available through the VM depends on the software role assigned to that VM during provisioning. A VM configured with application software may provide application server functionality, and that a VM configured with database-server software may provide access to a database, because those are ordinary software roles for a virtual machine in the type of provisioning environment Khandekar describes. *Id.*, 11:50-58, 23:11-20.

156. Bandhole is consistent with that understanding because Bandhole describes remotely accessible computing environments that include application servers, middleware-tier functionality, and database-related components. MS-1007, [0042]-[0045], [0048], [0051]-[0054]. Bandhole describes a multi-tier architecture in which server-side components receive requests from web clients and perform processing associated with application-server and middleware-type roles. *Id.* A POSITA would have understood from Bandhole that application server and database-oriented functions were well-known roles for remotely accessed virtualized or server-side resources.

C. Ground 1D: Claims 6-8 Are Obvious Over The Combination of Khandekar And Vasilevsky

1. Claim 6

[6] The method of claim 1, wherein the virtual asset is providing a virtual storage resource.

157. The Khandekar-Vasilevsky System would have provided a VM (*virtual asset*) configured to include, utilize, or present a virtualized storage construct (e.g., “virtual storage entity”), which provides an example of a *virtual storage resource*. MS-1004, 6:24-42; MS-1005, [0150], [0156], [0172].

158. As I discussed in Ground 1, Khandekar’s VM includes a “virtual disk” (e.g., 314) that stores the VM’s operating system and other installed software. A POSITA would have recognized and/or found obvious that the VM may be implemented to utilize advanced virtual storage configurations, such as the virtualized resources described in Vasilevsky. In Vasilevsky, a virtual resource includes a “virtual storage entity” that is distinct from an “actual storage device” (e.g., Vasilevsky’s “disks 914”). MS-1005, [0166], Cl. 80. In the Khandekar-Vasilevsky System, the VM (*virtual asset*) provides a “virtual storage entity” as a *virtualized storage resource* presented as part of, or to support, the virtualized server instance.

2. *Claim 7*

[7] The method of claim 6, wherein the virtual storage resource further comprises mapping a logical storage unit to a physical storage disk.

159. As I discussed for [6], the Khandekar-Vasilevsky System would have provided a VM (*virtual asset*) that includes a virtualized storage construct, such as a “virtual storage entity” corresponding to physical storage. Consistent with Vasilevsky, in the combination, the virtual storage construct includes a mapping between (i) a storage abstraction presented logically to a virtualized system (*logical storage unit*) and (ii) a hardware component (*physical storage disk*). MS-1005, Cls. 78, 84, 85.

160. Vasilevsky describes maintaining a “mapping” between a “virtual resource” and “actual resources” to enable allocation and management of resources for virtualized computing instances. MS-1005, Cl. 78. The actual resources include hardware components, such as “one or more storage devices (e.g., disks 914).” *Id.*, [0166]. The system also includes a “mapping database that defines mapping of virtual resources to physical hardware.” *Id.*, [0150]. A POSITA would have understood that, for storage resources, the mapping database provides the association between the logical storage abstraction exposed at the virtual layer and the particular underlying hardware disk(s) used to implement that abstraction. The ’915 patent specification likewise describes mapping a logical storage construct (e.g., a logical storage unit) onto underlying storage hardware (e.g., disk drives or

storage devices). See MS-1001, 20:45-21:3.

3. *Claim 8*

[8] The method of claim 7, wherein providing the virtual storage resource further comprises mapping a logical storage unit to a physical storage disk.

161. See [7], *supra*.

4. *Claim 11*

[11.pre] A method, comprising:

162. See my discussion of [1.pre], above.

[11.1] communicating, by a user, over a network to access one or more virtual assets running in a virtualization system,

163. See my discussion of [1.pre], above. The Khandekar-Vasilevsky System would have enabled a user (e.g., 900) to communicat[e] over a network (e.g., network 700) (***network***) with the provisioning server and portal interface to access one or more VMs (e.g., VMx, VMy) (***virtual assets***) operating on host servers in the virtualization environment (***virtualization system***). MS-1004, 9:46-64, 13:66-14:3, 14:4-18.

[11.2] wherein the virtualization system includes a control computer and one or more physical resources,

164. The Khandekar-Vasilevsky System would have provided a virtualization system including a centralized provisioning server with a Hirshfeld-like SCM (control computer) that is coupled over a network to one or more host servers (e.g., 1000, 1001) that provide underlying hardware infrastructure (one or

more physical resources). MS-1004, 9:26-32, 9:33-45, 9:46-64, 10:27-42.

165. Khandekar explains that a provisioning server (e.g., 800) manages a plurality of host computers on which virtual machines are deployed. MS-1004, 9:26-32, 10:27-42. Khandekar further describes that a virtual machine monitor (VMM) is pre-installed on each host and virtualizes resources of the host machine while running directly on top of the host hardware. *Id.*, 6:63-7:6, 7:7-27, 13:17-35. In this configuration, the hardware infrastructure of the host servers correspond to ***physical resources*** because they supply resources that are virtualized and used to execute virtualized instances under centralized control. *Id.*, 7:7-27, 10:27-42.

166. Additionally, Khandekar describes that the provisioning server communicates with the host machines via a network and coordinates provisioning operations on selected hosts, reflecting that the control computer manages and communicates with the one or more physical resources in the virtualization system. MS-1004, 9:46-64, 10:27-42, 21:51-52.

[11.3] wherein each of the one or more physical resources execute at least two virtual assets that comprise software instantiations of computer systems and

167. The Khandekar-Vasilevsky System would have provided host servers (e.g., 1000, 1001) (***physical resources***) that each execute at least two VMs (***virtual assets***), where each VM represents an example of a ***software instantiation[] of computer systems***. MS-1004, 5:37-49, 6:24-42.

168. Khandekar explains that a VM is a “software abstraction” of a “physical computer system,” and further describes software constructs implemented within the VM to emulate components of an actual computer, including virtual peripheral devices implemented in software and device emulators that emulate characteristics of physical devices. MS-1004, 6:24-42, 7:48-55. Khandekar also states that “more than one virtual machine can normally be installed on a single host server,” which a POSITA would understand as satisfying (at minimum) the requirement that each physical host execute at least two virtual assets. MS-1004, 5:37-49.

[11.4] the virtual assets are managed by virtualization layer instances running on each of the one or more physical resources;

169. The Khandekar-Vasilevsky System would have provided host servers (e.g., 1000, 1001) (***one or more physical resources***) that each execute a respective virtual machine monitor (VMM) as a virtualization layer instance, where that VMM manages the virtual machines (***virtual assets***) executing on that host. MS-1004, 6:63-7:6, 7:7-27, 13:17-35.

170. Khandekar describes that a VMM is “pre-installed on each host” within a plurality of hardware hosts, and that the VMM “virtualizes all, or at least some of, the resources of the machine” while running “directly on top of the host.” MS-1004, 6:63-7:6, 7:7-27, 13:17-35. A POSITA would understand that because each host executes its own VMM instance to virtualize the host’s resources for the

VMs running on that host, the VMM on each physical resource is a virtualization layer instance that manages execution of the virtual assets on that physical resource by providing the virtualized view of underlying host resources. MS-1004, 7:7-27.

[11.5] requesting, by the user, provisioning of a virtual asset of the one or more virtual assets in the virtualization system; and

171. The Khandekar-Vasilevsky System would have received, from a user communicating via a console (e.g., 801) (***requesting, by the user***), a request to provision a selected VM (e.g., VMx) (***provisioning of a virtual asset***) within the system (***virtualization system***). MS-1004, 9:46-64, 13:66-14:3, 19:63-20:2, 20:64-21:8.

172. As I discussed for [1.4] and [1.5], Khandekar describes that an end user interacts with provisioning server 800 through user console 801 using “any conventional web browser.” MS-1004, 9:46-64, 13:66-14:3. Khandekar then describes what happens when a provisioning request is submitted: “[w]hen the user submits a request to provision a new VM, the provisioning engine requests that a new task be created,” and the task manager returns a task identifier for the requested provisioning operation. *Id.*, 20:64-21:8. The user request can be for a pre-configured VM or can include user-supplied configuration inputs (e.g., “computer hostname, IP address, etc.”) submitted through the console connection. *Id.*, 13:66-14:3, 19:63-20:2. Thus, in the Khandekar-Vasilevsky System, the user may submit (and the system receives) a request to provision a virtual asset (a new

VM) in the virtualization system.

[11.6] wherein the provisioned virtual asset is a virtual storage and the virtual storage comprises one or more logical storage units mapped to one or more physical storage drives.

173. In the Khandekar-Vasilevsky System, a selected VM provisioned based on a user provisioning request (***provisioned virtual asset***) may be configured to function as a storage server for the network. This allows the VM to act as an example of a ***virtual storage resource***.

174. A POSITA would have understood that VMs are commonly provisioned specifically to deliver storage services. Vasilevsky, for example, describes replacing specialized storage hardware with a “software-only application that is executed on a virtual server,” including by providing “Network Attached Storage (NAS) functions.” MS-1005, [0011]. Consistent with that teaching, in some implementations of the Khandekar-Vasilevsky System, the provisioned VM functions as a ***virtual storage***.

175. To provide storage capabilities, the provisioned VM may also utilize underlying virtualized storage constructs. Consistent with Vasilevsky, the system manages these constructs by maintaining a mapping between (i) a storage abstraction presented logically to the virtualized system (e.g., Vasilevsky’s “virtual storage entity”) (***one or more logical storage units***) and (ii) underlying hardware storage devices (e.g., Vasilevsky’s “disks 914”) (***one or more physical storage***

drives). MS-1005, [0150], [0166], Cls. 78, 80. In such configurations, the provisioned VM operates as an example of *virtual storage* in that it maps logical storage to physical storage drives.

5. *Claim 12*

[12] The method of claim 11 further comprising requesting, by a management application programming interface (API) of the virtualization system, the provisioning of the virtual storage.

176. The Khandekar-Vasilevsky System would have enabled a user to request provisioning operations through a programmatic interface of the provisioning server (e.g., 800) (*application programming interface (API) of the virtualization system*). This includes provisioning operations directed to storage components of a provisioned VM (*provisioning of the virtual storage*). MS-1004, 9:46-64, 22:53-67.

177. Khandekar describes that provisioning requests may be submitted not only by a human user, but also by “a software application or tool.” MS-1004, 9:46-64. In such implementations, “[s]oftware applications acting as a user interact with the provisioning server using a well-defined programmatic interface (API)...” *Id.* Thus, in implementations of the Khandekar-Vasilevsky System, provisioning requests are issued via an API that forms part of the provisioning server's control-plane interface for provisioning operations. *Id.*, 9:46-64, 22:53-67.

178. As I discussed for [11.6], the Khandekar-Vasilevsky System would

have enabled provisioning of a selected VM that includes a virtualized storage construct, such as a “virtual storage entity” corresponding to physical storage (*virtual storage*).

6. *Claim 13*

[13] The method of claim 12 further comprising configuring, by the user, the virtual storage.

179. As I discussed above for [6], in the Khandekar-Vasilevsky System, user-selected configuration inputs would have included selections for the VM’s virtualized storage construct. Consistent with Vasilevsky, a POSITA would have understood that these inputs may be used to configure a VM’s virtual storage, including configuring a virtual storage entity and its associated mapping to underlying actual storage devices as part of the system’s management and provisioning operations. MS-1005, [0040], [0150], [0166]. In implementations where the user specifies storage parameters (e.g., virtual disk capacity or related storage settings) during VM provisioning, this involves the user *configuring* the configuring the *virtual storage*, and the Khandekar-Vasilevsky System implements that configuration using Vasilevsky’s virtual storage management framework. MS-1005, [0040], [0150].

7. *Claim 14*

[14] The method of claim 13 further comprising receiving, by the user, an inventory of the virtual storage.

180. The Khandekar-Vasilevsky System would have enabled a user (via a

management-plane interaction used for requesting provisioning) to receive a listing of available virtual storage resources (*inventory of the virtual storage*) presented for selection/assignment in connection with provisioning operations. MS-1004, 10:15-26, 11:1-10, 13:66-14:3, 15:29-39; MS-1005, [0084], [0150].

181. As I discussed for [13], Khandekar provides a user-facing management pathway (e.g., a user console accessed via a web browser) through which a user submits requests and receives system-provided information (e.g., selectable options and task-related feedback). MS-1004, 10:15-26, 11:1-10, 13:66-14:3, 15:29-39. Vasilevsky describes that a manager may “perform an association” between “unallocated resources” and virtual servers, and that this configuration access can be performed by “multiple entities (administrators, programs, etc.).” MS-1005, [0084], Cl. 84. In the context of Vasilevsky's virtualization management, this association of unallocated resources would include storage-related virtual resources (e.g., virtual storage entities) corresponding to underlying physical storage devices under the system's mapping/assignment framework. *Id.*, [0150], Cls. 78, 80. In the Khandekar-Vasilevsky System, when the user (or a management interface acting on the user's behalf) requests provisioning of virtual storage, the system would present and the user would receive an inventory of available virtual storage resources (e.g., available virtual storage entities/logical storage units available for assignment) to support the requested provisioning.

D. Ground 1E: Claims 9 And 10 Are Obvious Over The Combination of Khandekar And Fujino

1. Claim 9

[9] The method of claim 1, wherein providing the virtual assets further comprise providing a logical network resource.

182. The Khandekar-Fujino System would have provided a provisioned VM (*virtual asset*) configured to utilize a software-defined representation of network connectivity (e.g., SAN configuration information) (logical network resource) that specifies connectivity between host computers and storage systems by identifying physical devices along the communication path (e.g., fiber channel switches and endpoint ports). MS-1004, 9:26-32, 10:50-60; MS-1008, 6:27-36, 10:16-30.

183. As I discussed for [1.1]-[1.5], Khandekar describes provisioning and providing user access to VMs (e.g., VMx, VMy) deployed on host servers (e.g., 1000, 1001) under control of a provisioning server. MS-1004, 9:26-32, 10:50-60. In the Khandekar-Fujino System, the combination would have further incorporated Fujino's Storage Area Network (SAN) architecture to provide network connectivity between these host servers and underlying storage devices. MS-1008, 5:31-45. Fujino describes that "[t]he SAN 19 may be constituted by a plurality of fiber channel switches (FC-SW hereafter) 71," which are physical devices on the network used to route communications. *Id.*, 6:27-36.

184. Fujino also describes maintaining SAN configuration information that

includes, for example, “an ID for the port of the storage control device 27, and one or more IDs corresponding respectively to the one or more FC-SWs existing between these ports.” MS-1008, 10:16-30. A POSITA would have understood and/or found it obvious that this SAN configuration information is a logical (software-defined) abstraction that identifies underlying physical network devices (ports, switches) to define the communication path used by the system, and thus, represents an example of a **logical network resource**. MS-1008, 10:16-30.

2. *Claim 10*

[10] The method of claim 9, wherein providing the logical network resource further comprises mapping a logical network unit to a physical network device.

185. The Khandekar-Fujino System would have provided a software-defined network path representation (e.g., maintained in the SAN configuration information) (**logical network unit**) that is associated with (**mapp[ed]**) to the underlying network hardware (**physical network device**), such as the physical ports of the storage control device and the intervening fiber channel switch hardware used to implement the communication path. MS-1008, 6:27-36, 6:40-49, 7:35-43, 10:16-30.

186. As I discussed for [9], Fujino describes maintaining SAN configuration information that defines connectivity between endpoints by identifying physical devices along the path, including “an ID for the port of the

storage control device 27, and one or more IDs corresponding respectively to the one or more FC-SWs existing between these ports.” MS-1008, 10:16-30; *see also id.*, 7:35-43. Fujino also describes that the storage control device includes “one or more” physical “ports” for connecting to the SAN, which are tangible network hardware devices. MS-1008, 6:40-49. The SAN configuration information is an example of the ***logical network unit*** because it represents network connectivity as a software-defined abstraction that identifies (and thereby maps to) underlying physical network devices (ports, switches). For instance, the information specifies identifiers that correspond to (and thereby specify) the particular physical port(s) and fiber channel switch(es) that implement the communication path. MS-1008, 6:27-36, 6:40-49, 10:16-30.

E. Ground 2A: Claims 1 And 2 Are Obvious Over The Combination of Hirschfeld And Smith

1. Claim 1

[1.pre] A method for access to one or more virtualization system resources over a network, the method comprising:

187. The Hirschfeld-Smith System would have satisfied the requirements of [1.pre]. Hirschfeld describes a “method of virtualizing a plurality of logical servers into a logical server cloud linked to a plurality of physical servers.” MS-1006, 4:54-5:3. In Hirschfeld, “a user of any remote terminal 105 may login to and access any of the logical servers 106 of the virtualized logical server cloud 101 via the user interface 104 and SCM 102 as though logging into a server directly or via

a local area network (LAN).” MS-1006, 6:54-63.

188. The remote terminals 105 communicate with the SCM and logical servers through an “intermediate network system 103,” which “may be of any known type, both publicly accessible networks such as the Internet and proprietary networks.” MS-1006, 6:28-42. The Hirschfeld-Smith System would have enabled access to one or more logical server resources (e.g., resources of logical servers 106 of the virtualized logical server cloud) (one or more virtualization system resources) over an intermediate network (e.g., 103) (network).

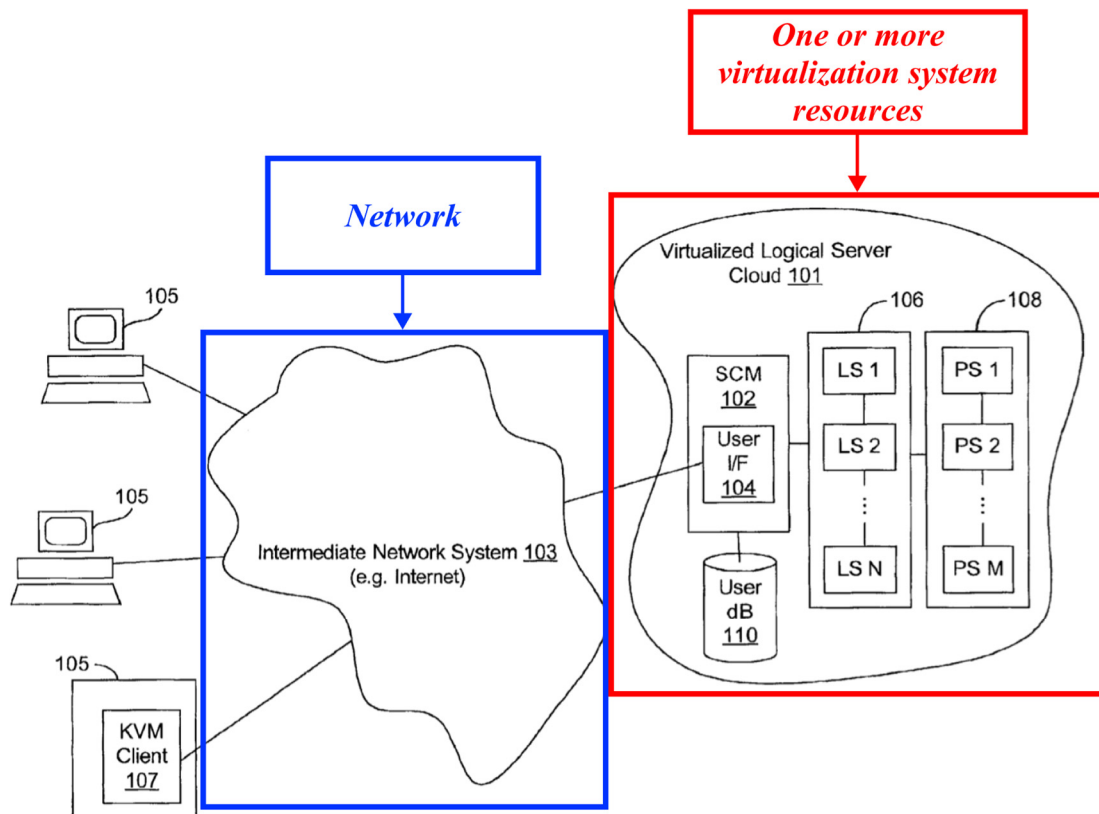


FIG. 1

MS-1006, FIG. 1

[1.1] implementing a virtualization system that has a control computer and one or more physical resources and virtual assets connected to the control computer,

189. The Hirschfeld-Smith System (*virtualization system*) would have provided an SCM (*control computer*) that interfaces with and controls a bank of physical servers (*one or more physical resources*) hosting logical servers (*virtual assets*), where the physical resources and virtual assets are connected to the SCM. MS-1006, 5:66-6:27, 6:34-7:5, 7:24-30.

190. As I discussed for [1.pre], Hirschfeld describes a “virtualized logical server cloud” that includes an SCM representing “cloud management infrastructure.” MS-1006, 5:66-6:27. The SCM “has the authority to allocate physical resources to maintain the logical server cloud” and thereby defines “physical resources under management control.” *Id.*, 6:7-15. Moreover, “the physical servers controlled by the SCM determine a logical server cloud's boundary.” *Id.* The SCM functions as a centralized management component that controls resource allocation for the virtualized logical server cloud. Because the SCM has authority over resource allocation and defines the boundary of the logical server cloud through the physical servers it controls, a POSITA would understand the SCM to be an example of a control computer as recited in the claim.

191. The virtualized logical server cloud includes “one or more logical servers 106” (*virtual assets*). MS-1006, 6:34-38. These logical servers are

connected to the SCM because, for example, the “SCM 102 provides KVM redirection either by providing a KVM interface or by mapping the KVM client 107 to allow direct KVM access to the selected logical server 106.” MS-1006, 6:64-7:5. The KVM redirection establishes a communication pathway between the SCM and each logical server, which reflects a connection between the virtual assets and the control computer.

192. The virtualized logical server cloud also includes “a bank of physical servers 108” (*one or more physical resources*). MS-1006, 6:38-42. The SCM is connected to the bank of physical servers because, for example, the user interface associated with the SCM “also provides interfaces to provide SCM control of the physical server to enable other functions including copy, move, start, shutdown, reset and any other functions enabled by control over the physical server and virtualization software.” *Id.*, 7:24-30. These interfaces establish a control relationship between the SCM and each physical server, which reflects a connection between the physical resources and the control computer.

193. In the Hirschfeld-Smith System, the SCM (*control computer*) is connected to and controls each of the bank of physical servers (*physical resources*) and the logical servers (*virtual assets*) instantiated on those physical servers. MS-1006, 6:48-54.

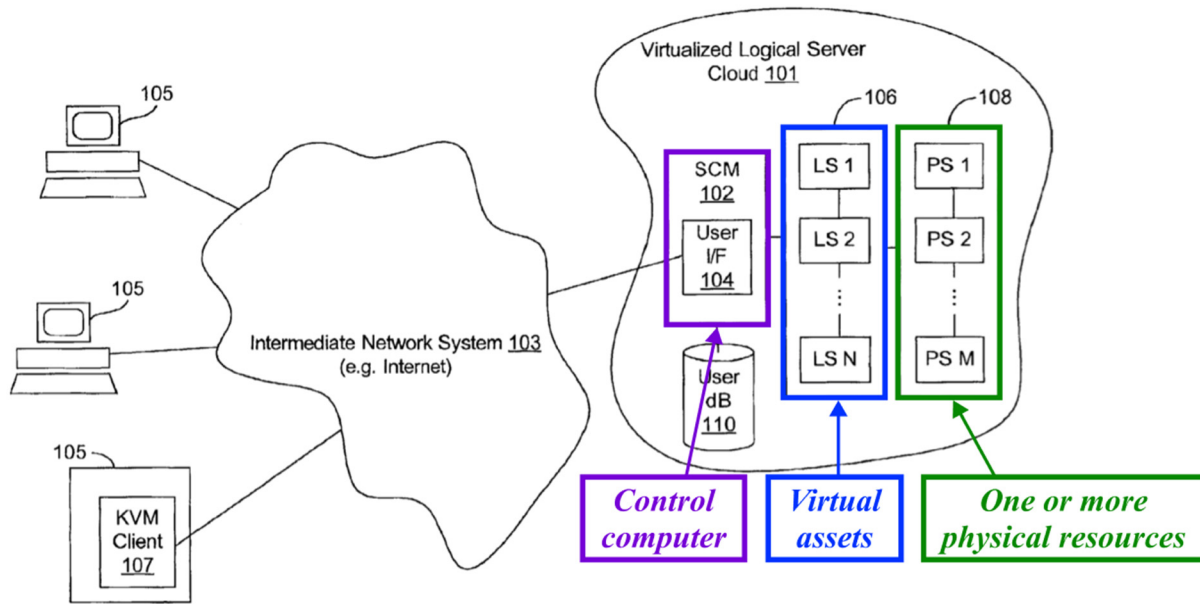


FIG. 1

MS-1006, FIG. 1

[1.2] wherein each of the one or more physical resources execute at least two virtual assets that comprise software instantiations of computer systems, and

194. The Hirschfeld-Smith System would have provided physical servers (*physical resources*) that each execute multiple logical servers (*at least two virtual assets*), where the logical servers are provided using virtualization technology and therefore are examples of software instantiations of computer systems. MS-1006, 1:63-2:7, 6:34-46, 8:1-10, 10:22-27, FIGS. 2, 3.

195. As I discussed for [1.1], Hirschfeld describes a virtualized logical server cloud that includes “one or more logical servers 106” and a “bank of physical servers 108,” where the number of physical servers (“M”) “may be a

smaller number" than the number of logical servers ("N"). MS-1006, 6:28-63, 8:1-10. When "M" is smaller than "N," at least one physical server necessarily executes multiple logical servers. This follows because if there are fewer physical servers than logical servers and each logical server is linked to a physical server, then at least one physical server must host more than one logical server (*at least two virtual assets*).

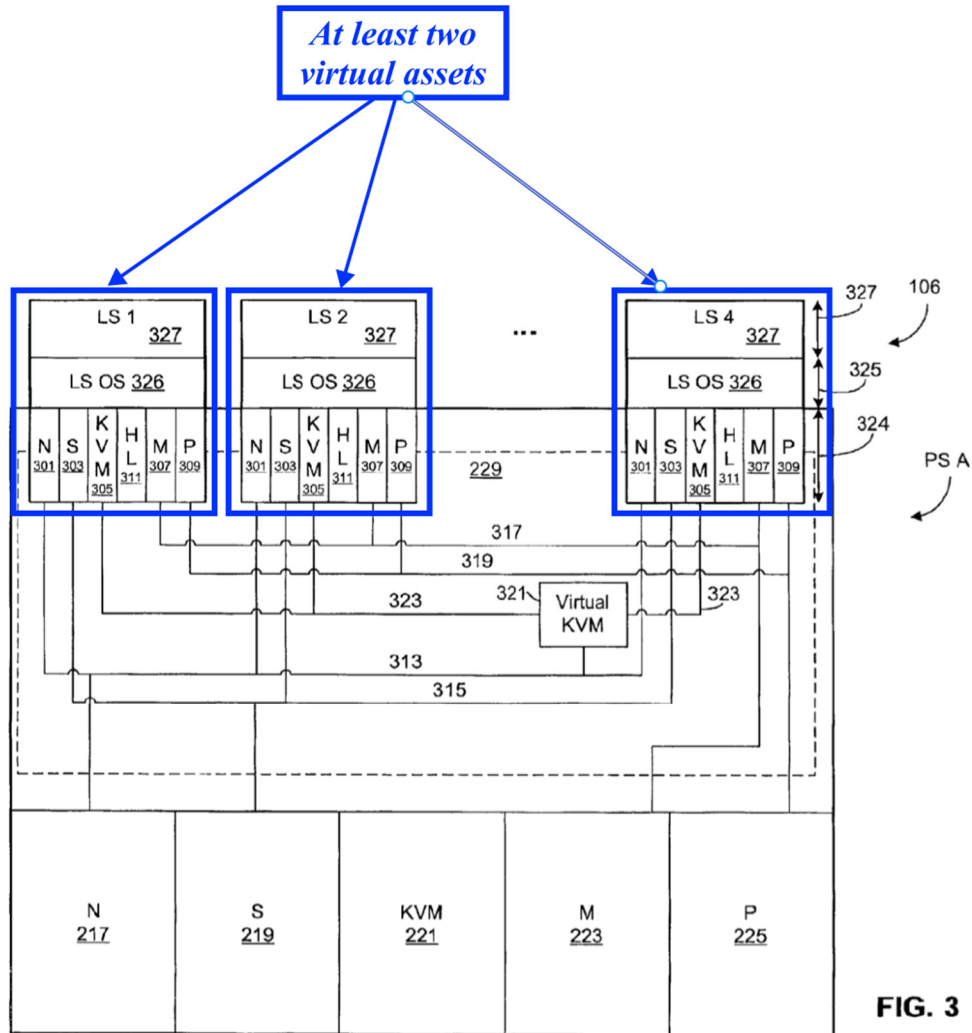
196. Hirschfeld confirms this understanding with specific examples. In Figure 2, a "logical server cloud" is formed with multiple logical servers linked to physical servers, e.g., "logical servers LS 1-4 are linked to the PS A, the logical servers LS 5-8 are linked to the PS B, and the logical servers LS 9-N are linked to the PS C." MS-1006, 6:28-63, 8:1-10. In this configuration, each physical server (PS A, PS B, PS C) executes multiple logical servers (e.g., PS A executes four logical servers (LS 1-4)).

197. Hirschfeld's Figure 3 provides another example, portraying multiple logical servers running on a single physical server. In this configuration, each logical server operating system "is executed independently from every other LS OS 326, or otherwise gives every appearance of independent operation from the user's perspective." MS-1006, 10:9-31. This independent execution on shared physical hardware is characteristic of virtualization technology in which each logical server operates as a self-contained software construct on the underlying

physical server.

198. Hirschfeld also describes that virtualization “allow[s] multiple logical servers to override a physical server's attributes and share its resources.” MS-1006, 2:1-7. A POSITA would have implemented the Hirschfeld-Smith System to leverage these techniques such that its physical servers (*one or more physical resources*) each execute at least two logical servers (*virtual assets*), e.g., physical server PS A running logical servers LS 1-4 and physical server PS B running logical servers LS 5-8. MS-1006, 8:1-10.

199. Hirschfeld’s logical servers are also examples of software instantiations of computer systems. Hirschfeld describes “the use of virtualization technology that enables multiple logical servers to operate on a single physical computer.” MS-1006, 1:63-2:7. Because virtualization technology creates logical servers as software constructs that override and share the physical server’s resources, each logical server is implemented in software as a virtualized computer system (*software instantiation of a computer system*). In the Hirschfeld-Smith System, each physical server (*physical resource*) executes at least two logical servers (*virtual assets*) that are software instantiations of computer systems. MS-1006, 1:63-2:7, 8:1-10.



MS-1006, FIG. 3

[1.3] *the virtual assets are managed by virtualization layer instances running on each of the physical resources;*

200. The Hirschfeld-Smith System would have provided physical servers (*physical resources*) that each execute virtualization software (e.g., a virtual machine monitor (VMM)) (*virtualization layer instances*) to manage logical servers (*virtual assets*) linked to and executing on that physical server. MS-1006, 9:35-57; MS-1010, 369.

201. As I discussed for [1.1] and [1.2], each physical server in Hirschfeld's bank of physical servers executes virtualization software. For example, “[e]ach physical server 108 executes virtualization software 229,” where “virtualization software 229 manages physical resources of the logical servers 106 linked thereto.” MS-1006, 9:35-57. The virtualization software 229 running on a given physical server manages the logical servers linked to that physical server (i.e., it is responsible for virtualizing the physical resources that the logical servers use). In this configuration, the virtualization software 229 running on a given physical server is an example of a virtualization layer instance running on that physical resource that manages the corresponding logical servers (*virtual assets*).

202. Although Hirschfeld describes virtualization software 229 as managing the physical resources of linked logical servers, Hirschfeld does not describe in detail the specific mechanisms by which the virtualization software performs this management. Smith provides relevant context on this point. In Smith, a VMM “manag[es] the allocation of, and access to, the hardware resources of the host platform,” “[t]he VMM owns the real system resources and makes them available to one or more guest operating systems that alternately execute on the same hardware,” and “a guest operating system in a virtual machine environment is given the illusion of owning the resources it subsequently allocates to its various user programs.” MS-1010, 369. A POSITA would have understood Smith’s

description of VMM functionality to be consistent with and illustrative of the type of virtualization management performed by Hirschfeld's virtualization software 229. This includes managing the allocation of physical server resources to logical servers such that each logical server operates as though it has its own dedicated resources.

203. As discussed above, in the Hirschfeld-Smith System, each physical server runs virtualization software implemented as a VMM (*virtualization layer instance*) that manages the logical servers (*virtual assets*) by managing allocation of and access to the physical server's hardware resources. MS-1006, 9:35-57; MS-1010, 369. This is consistent with Hirschfeld, which explains that "at least one physical server executes virtualization software that virtualizes physical resources for logical servers." MS-1006, Abst. Hirschfeld further explains that this virtualization "allow[s] multiple logical servers to override a physical server's attributes and share its resources." MS-1006, 2:1-7.

[1.4] connecting an end user to the control computer; and

204. The Hirschfeld-Smith System would have connected a user (end user) to the SCM (*control computer*) through a user interface (e.g., user interface 104), thereby providing a networked connection to the SCM for logging in and interacting with the virtualized logical server cloud. MS-1006, 6:28-63.

205. As I discussed for [1.pre], Hirschfeld describes that "a user of any

remote terminal” may “login to and access” logical servers via a user interface of the SCM “as though logging into a server directly or via a local area network (LAN).” MS-1006, 6:28-63. The user’s login and access to logical servers is mediated through the SCM. In other words, the user communicates with the SCM’s user interface, and the SCM provides access to the logical servers. This interaction establishes a network connection between the end user and the SCM (*control computer*), because the user must first connect to the SCM in order to reach the logical servers behind it. The Hirschfeld-Smith System would have therefore connected the user (*end user*) to the SCM (*control computer*).

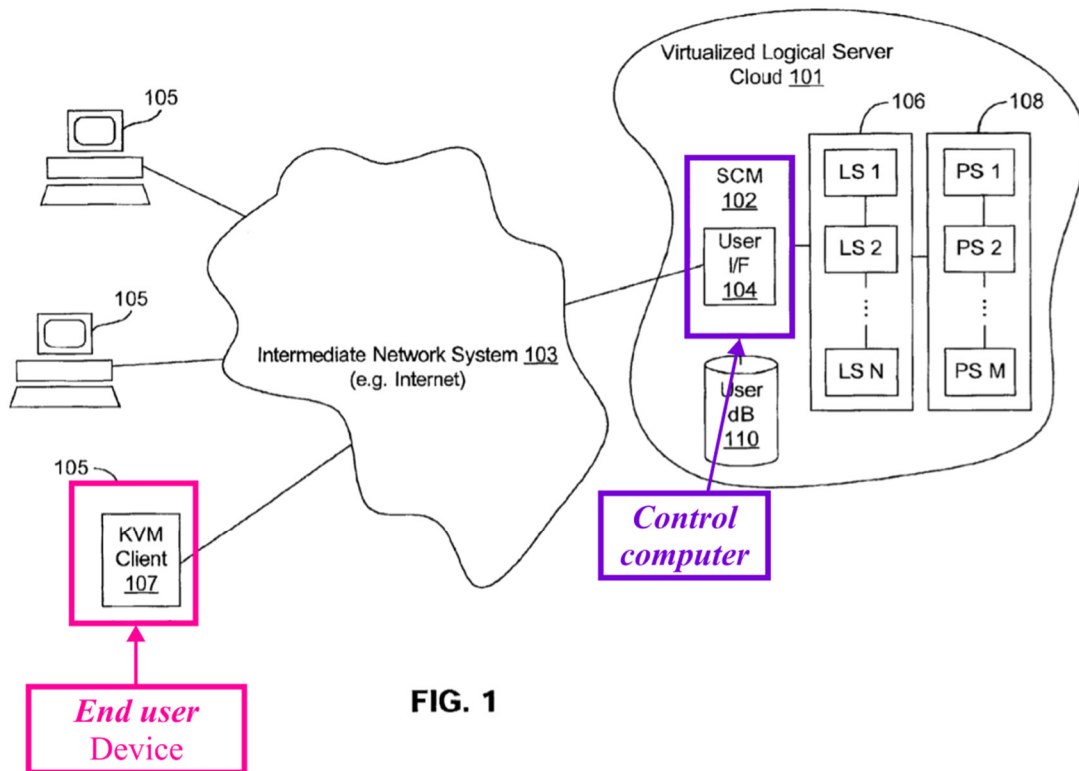


FIG. 1

MS-1006, FIG. 1

[1.5] accessing, through the control computer connection, the virtual assets by the end user.

206. The Hirschfeld-Smith System would have enabled the user to access the logical servers (*virtual assets*) implemented in the virtualized logical server cloud through the user's connection to the SCM (*control computer*) via a user interface. MS-1006, 6:28-63, 14:39-61.

207. As I discussed for [1.4], Hirschfeld describes that a user “may login to and access any of the logical servers” via a user interface of the SCM, e.g., “as though logging into a server directly or via a local area network (LAN).” MS-1006, 6:28-63. In this configuration, the user first connects to the SCM (as I discussed for [1.4]), and then accesses the logical servers (*virtual assets*) through that connection. The SCM thereby serves as the intermediary through which the user reaches the logical servers, such that the user's access to the logical servers is routed through the SCM (*control computer connection*).

208. Hirschfeld provides a specific mechanism by which this access is implemented. In particular, Hirschfeld describes the SCM providing KVM redirection such that “KVM requests to and from [a logical server] are handled via the KVM client 519 of the SCM 102.” MS-1006, 14:39-61. In this architecture, the SCM acts as a proxy that routes the user’s keyboard, video, and mouse communications to and from the target logical server. A POSITA would understand this KVM redirection as an example of the SCM providing a

communication pathway through which the user accesses the logical server, with the SCM functioning as the network-facing access point in the data path between the user and the logical server.

209. For example, when accessing a newly deployed VM, a POSITA would use Hirschfeld's proxy-style approach so that the remote-access session is routed through the provisioning server rather than requiring the user to connect directly to the underlying host. One example implementation Hirschfeld describes is Keyboard/Video/Mouse (KVM) redirection, where "KVM requests to and from [logical server (or VM)] are handled via the KVM client 519 of the SCM 102." MS-1006, 14:39-61. This type of proxy routing means the user's connection to the provisioning server serves as the pathway for accessing the provisioned VM, i.e., an example of a control computer connection enabling the user to access the provisioned VM.

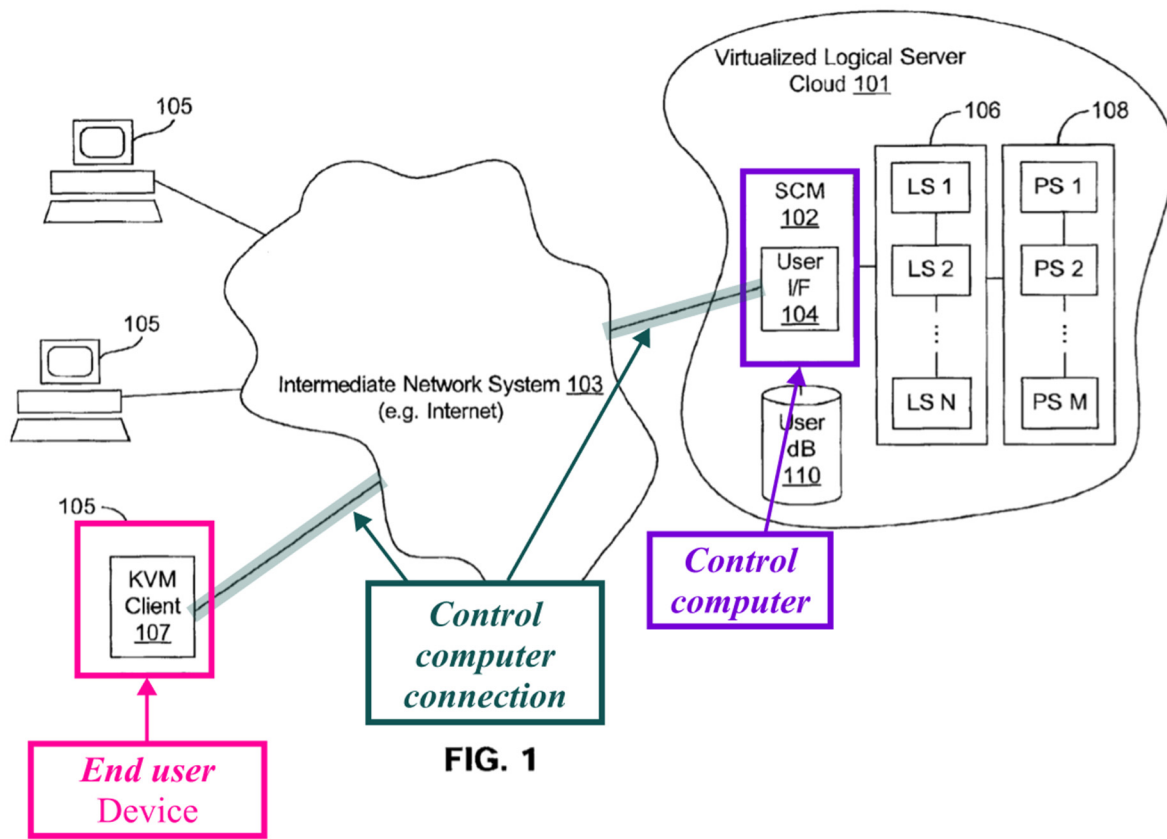


FIG. 1

MS-1006, FIG. 1

2. Claim 2

[2] *The method of claim 1, wherein connecting the end user to the control computer further comprises connecting the end user to the control computer over one of a network, a wide area network or the Internet.*

210. The Hirschfeld-Smith System would have connected a user (*end user*) to the SCM (*control computer*) over a network, including implementations where the network is the Internet. MS-1006, 6:28-63.

211. As I discussed for [1.4], the user connects to the SCM through a user interface to access logical servers. Hirschfeld describes that the virtualized logical

server cloud is “accessible to one or more remote terminals” via an “intermediate network system,” such as “the globally-accessible Internet, although any type of network is contemplated.” MS-1006, 6:28-63. The SCM includes a user interface that “provides a web page portal or other access mechanism to facilitate communication with the remote terminals” via that network system. *Id.* Because the user accesses the SCM’s user interface through the intermediate network system (which may be the *Internet*), the user’s connection to the SCM (*control computer*) is established over a *network*, a *wide area network*, or the *Internet*.

F. Ground 2B: Claims 3-5 Are Obvious Over The Combination of Hirschfeld, Smith, And Bandhole

1. Claim 3

[3] The method of claim 2, wherein connecting the end user to the control computer further comprises providing the end user, a desktop, a laptop or a personal digital assistant (PDA) and client resources.

212. The Hirschfeld-Smith-Bandhole System would have permitted a user to access logical servers (virtual assets) through a connection established using a user interface of the SCM (control computer) (connecting the end user to the control computer), and a POSITA would have understood that the user device used for that access may be one of several well-known device types, including a “workstation” (desktop, client), “laptop” (laptop), and “palmtop” (personal digital assistant (PDA)). MS-1006, 6:28-63, 7:32-48; MS-1007, [0035].

213. As discussed for [1.5], Hirschfeld describes that a user accesses

logical servers through the SCM's user interface, including by logging in from a remote terminal and accessing logical servers for which the user has permission. MS-1006, 6:28-63, 7:32-48. Hirschfeld also describes that the user-side terminals may be configured with different levels of capability, including "simple terminals, thin client computers, low end PCs, etc." Id., 6:28-63. A POSITA would have understood from that disclosure that Hirschfeld is not limited to any single specialized endpoint device, but instead contemplates a range of client-side devices that can communicate with the SCM over the network. A POSITA would have also understood that such devices necessarily include client-side resources, such as local computing resources and network-interface resources, because those are the resources that allow the device to run the client-side access mechanism and communicate with the SCM's user interface.

214. Bandhole is consistent with that understanding because Bandhole describes remotely accessible computing environments in which client devices communicate over a network with server-side components and explains that such computing devices may come in many "dimensions and styles," including "workstation," "laptop," and "palmtop" form factors. MS-1007, [0015], [0035], [0048]. A POSITA would have understood Bandhole to confirm that these form factors were well-known and suitable implementations for accessing and managing remote virtual computing environments through network-facing interfaces. A

POSITA would have also understood that Bandhole's listed form factors map naturally onto the claim language reciting a desktop, a laptop, or a PDA because a workstation is a desktop-class client device, a laptop is expressly identified, and a palmtop is a handheld client device corresponding to a PDA-style platform. Id., [0035].

2. *Claim 4*

[4] The method of claim 3, wherein accessing the virtual assets further comprising accessing an application server by the end user.

215. The Hirschfeld-Smith-Bandhole System would have permitted a user to access logical servers (virtual assets) through a connection established using a user interface of the SCM (accessing the virtual assets), and a POSITA would have understood that, in some implementations, the accessed logical server provides application server functionality. MS-1006, 10:9-31; MS-1007, [0005], [0009], [0044], [0048], [0051]-[0054].

216. As discussed for [2] and [1.5], Hirschfeld describes a virtualization environment in which a user accesses logical servers through the SCM and associated user interface, and those logical servers operate as virtualized computing systems with their own operating-system environments. MS-1006, 6:28-63, 10:9-31. Hirschfeld explains that each logical server operating system is executed independently from every other logical server operating system, or otherwise gives the appearance of independent operation from the user's

perspective. Id., 10:9-31. A POSITA would have understood from that description that Hirschfeld's logical servers are not limited to a narrow predefined function, but instead are virtualized server instances capable of executing software in the same general way as independently operated server systems. A POSITA would have also understood that such logical servers could be configured to perform conventional server roles, including application-serving roles, depending on the software deployed in the logical server environment.

217. Bandhole is consistent with that understanding because Bandhole describes remotely accessible computing environments that include multi-tier architectures with expressly identified "application servers," such as application server 311 and application servers 221 and 223. MS-1007, [0005], [0009], [0044], [0048], [0051]-[0054]. Bandhole explains that those application servers receive requests from web clients, such as browsers, and return results to the client through the server-side computing environment. Id. A POSITA would have understood from Bandhole that application-server functionality was a well-known and expected role for remotely accessed server-side computing resources in networked environments of this kind. A POSITA would have also understood that applying that known application-server role to Hirschfeld's logical servers would have been technically straightforward because Hirschfeld already provides isolated logical server instances capable of executing software while being accessed through a

centralized management interface.

3. *Claim 5*

[5]³ The method of claim 1, wherein the virtual asset is providing end user access to one of an application server, to a middleware server and to a database.

218. The Hirschfeld-Smith-Bandhole System would have enabled a user to access a logical server (virtual asset) through a connection to the SCM (control computer), and in some implementations that logical server would provide end user access to an application server, a middleware-type server function, or a database. MS-1006, 6:28-63, 10:9-31; MS-1007, [0042]-[0045], [0044], [0048], [0051]-[0054].

219. As discussed for [4], Hirschfeld describes logical servers as virtualized computing systems that execute software while maintaining virtual independence from one another. MS-1006, 10:9-31. Hirschfeld also describes that users access those logical servers through the SCM and associated user interface. Id., 6:28-63. A POSITA would have understood from that architecture that the functionality presented to the user through a logical server depends on the software role assigned to that logical server, because Hirschfeld's logical servers are

³ Because of the phrase “one of,” satisfaction of claim 5 does not mandate providing end user access to each of the enumerated elements recited in the claim.

software-executing virtualized server environments rather than fixed-function hardware appliances. A POSITA would have also understood that a logical server configured with application-serving software may provide application-server functionality to the end user, which is consistent with the same application-server reasoning discussed in [4]. Id., 10:9-31.

220. Bandhole is consistent with that understanding because Bandhole describes remotely administered computing environments that include application servers receiving requests from web clients and returning results to those clients. MS-1007, [0044], [0048], [0051]-[0054]. A POSITA would have understood from Bandhole that application-server functionality was a known and expected role for remotely accessed server-side computing resources in environments of this kind. A POSITA would have also understood that, when a Hirschfeld logical server is configured to perform that role, the virtual asset provides end user access to an application server through the user's connection to the SCM and the logical server environment.

221. Bandhole also describes a multi-tier architecture that includes a middleware tier containing one or more application servers and performing delegation of requests to services responsible for actions such as accessing a database, storage, or computing devices. MS-1007, [0042]-[0045]. A POSITA would have understood that this is an example of middleware-server functionality

because the disclosed tier sits between client-side requests and backend resources, coordinates request handling, and delegates operations to other services or resources in the system. A POSITA would have also understood that a logical server in the Hirschfeld-Smith-Bandhole System could be configured to provide that same kind of middleware function, given Hirschfeld's disclosure of software-executing logical server instances and Bandhole's disclosure of middleware-tier operations in remotely accessed server environments.

222. Bandhole further describes database-related functionality, including a data repository and database access within a three-tier architecture. MS-1007, [0009], [0051]-[0054]. A POSITA would have understood from that disclosure that, in some implementations, user access to the virtual asset may involve end user access to a database, including through application-server or middleware-server components that access and interact with the database on the user's behalf. A POSITA would have also understood that the claim is satisfied where the virtual asset provides end user access to any one of the recited alternatives. In the Hirschfeld-Smith-Bandhole System, that requirement is met at least in implementations where the logical server provides access to an application server, middleware-server functionality, or a database.

G. Ground 2C: Claims 6-8 And 11-14 Are Obvious Over The Combination of Hirschfeld, Smith, And Vasilevsky

1. Claim 6

[6] The method of claim 1, wherein the virtual asset is providing a virtual storage resource.

223. The Hirschfeld-Smith-Vasilevsky System would have provided a logical server (*virtual asset*) that further provides a virtualized storage construct (*virtual storage resource*), such as a virtualized storage resource of the logical server that is virtualized to an underlying physical server storage resource. MS-1006, 8:66-9:20; MS-1005, [0166], Cl. 80.

224. In Hirschfeld, each logical server includes a “storage resource” that is virtualized to the corresponding storage resource of the physical server. MS-1006, 8:66-9:20. Rather than having direct access to a physical disk device, the logical server’s storage resource is a virtualized construct that maps to the underlying physical storage through the virtualization software. A POSITA would understand that this virtualized storage resource (which presents a storage interface to the logical server while the actual storage resides on the physical server) is an example of a *virtual storage resource*.

225. Vasilevsky provides additional context confirming that virtual storage resources were a well-known aspect of virtualized environments. Vasilevsky describes a virtual resource that includes a “virtual storage entity” that is distinct from an “actual storage device” (e.g., “disks 914”). MS-1005, [0166], Cl. 80. A

POSITA would recognize that Vasilevsky's "virtual storage entity" and Hirschfeld's virtualized storage resource of a logical server describe the same concept (e.g., a storage resource presented to a virtualized server instance that is backed by, but distinct from, the underlying physical storage hardware).

2. *Claim 7*

[7] The method of claim 6, wherein the virtual storage resource further comprises mapping a logical storage unit to a physical storage disk.

226. As I discussed for [6], the Hirschfeld-Smith-Vasilevsky System would have provided a logical server (*virtual asset*) that includes a virtualized storage construct (*virtual storage resource*), such as a storage resource of the logical server that is virtualized to an underlying physical server storage resource. MS-1006, 10:32-56.

227. In Hirschfeld, the virtualization of a logical server's storage resource involves a mapping between the logical server's storage and the physical server's storage. Specifically, "the storage resource 303 of every logical server 106 is virtualized to the corresponding storage resource 219 of the underlying physical server." MS-1006, 10:32-56. This virtualization establishes a relationship between a storage abstraction presented to the logical server (*logical storage unit*) and the physical storage hardware of the underlying physical server (*physical storage disk*).

228. Vasilevsky confirms that such mappings between logical and physical

storage were a well-known aspect of virtualized environments. Vasilevsky describes a management agent that “may maintain a mapping database that defines mapping of virtual resources to physical hardware.” MS-1005, [0150]. Vasilevsky further describes that “the virtual resource is a virtual storage entity, and the actual resource includes an actual storage device.” MS-1005, Cl. 80. In this architecture, the manager “is adapted to perform a mapping between the virtual resource and the one or more actual resources,” including performing “an association between the one or more actual resources and at least one of the distributed server and virtual server.” MS-1005, Cls. 78, 84, 85. A POSITA would understand that these disclosed mappings between virtual storage entities and actual storage devices describe the same type of logical-to-physical storage mapping that Hirschfeld's virtualized storage resources implement.

3. *Claim 8*

[8] The method of claim 7, wherein providing the virtual storage resource further comprises mapping a logical storage unit to a physical storage disk.

229. The Hirschfeld-Smith-Vasilevsky System would have satisfied the requirements of [8] based on the same teachings I discussed for [7]. In particular, the virtualized storage construct in the Hirschfeld-Smith-Vasilevsky System includes a mapping between a logical server's virtualized storage resource (*logical storage unit*) and the physical server's storage resource (*physical storage disk*), as I

described for [7].

4. *Claim 11*

[11.pre] A method, comprising:

230. The Hirschfeld-Smith-Vasilevsky System satisfies the requirements of [11.pre]. Hirschfeld describes a virtualized logical server cloud in which a user at a remote terminal communicates over a network to log in to, access, and interact with logical servers through the SCM and associated user interface. MS-1006, 6:28-63. The combined system would have provided a user with network-based interaction with the virtualization environment to access one or more virtual assets.

[11.1] communicating, by a user, over a network to access one or more virtual assets running in a virtualization system,

231. The Hirschfeld-Smith-Vasilevsky System would have enabled a user to communicate over a network (*network*) with the SCM 102 to access one or more logical servers 106 (*virtual assets*) operating on physical servers 108 in the virtualized logical server cloud (*virtualization system*). MS-1006, 6:28-63.

Hirschfeld describes that “a user of any remote terminal 105 may login to and access any of the logical servers 106 of the virtualized logical server cloud 101 via the user interface 104 and SCM 102,” thereby disclosing user communication over a network to access virtual assets running in the virtualization system. Id.

232. As discussed for [11.pre], Hirschfeld describes a virtualized computing arrangement in which remote terminals communicate with the SCM

and user interface through an intermediate network to reach logical servers in the logical server cloud. MS-1006, 6:28-63. Hirschfeld also describes that the intermediate network may be the Internet and that the user accesses the logical servers through the SCM “as though logging into a server directly or via a local area network (LAN).” Id. A POSITA would have understood from that architecture that the user’s communication path is network-based because the remote terminal, intermediate network, user interface, and SCM collectively provide the communication framework through which the user reaches the logical servers hosted in the virtualized environment. A POSITA would have further understood that the logical servers are the resources being accessed through that network communication because Hirschfeld describes the user as logging in to and accessing logical servers rather than merely accessing an abstract management screen. Id.

233. Vasilevsky is consistent with that understanding. MS-1005, Abstract, [0016]-[0017], [0074]-[0079]. Vasilevsky describes a virtualized data-center environment in which virtual servers execute on underlying physical nodes and are managed through a management infrastructure and interface. Id. A POSITA would have understood that such a management interface permits user communication over a network to access and manage virtual assets running in the virtualization system because the disclosed management infrastructure is used to allocate

resources to virtual servers, to control those virtual servers, and to provide interfaces through which administrators or programs interact with the system. *Id.* A POSITA would also have understood that this is the same general type of network-based access model reflected in Hirschfeld, where a user communicates with management-side components to reach virtualized server instances executing on physical hardware.

[11.2] wherein the virtualization system includes a control computer and one or more physical resources,

234. The Hirschfeld-Smith-Vasilevsky System includes a SCM 102 (***control computer***) and one or more physical servers 108 (***one or more physical resources***) within the virtualized logical server cloud. MS-1006, 5:66-6:27, 6:34-42. Hirschfeld describes that the SCM provides the cloud management infrastructure for the virtualized logical server cloud and that the cloud includes a bank of physical servers. *Id.*

235. As discussed for [1.1], Hirschfeld describes that the SCM “has the authority to allocate physical resources to maintain the logical server cloud” and that the “physical servers controlled by the SCM determine a logical server cloud’s boundary.” MS-1006, 6:7-15. A POSITA would have understood from that description that the SCM is an example of the ***control computer*** because it performs centralized control functions for the virtualized environment. This includes allocation and management of the physical resources used to support the

logical server cloud. A POSITA would also have understood that the bank of physical servers is an example of *one or more physical resources* because those servers provide the underlying hardware on which the logical servers operate. *Id.*, 6:34-42.

236. Hirschfeld also describes that the SCM is connected to, and interacts with, both the logical servers and the physical servers through the cloud-management architecture. MS-1006, 6:48-54, 6:64-7:5, 7:24-30. That architecture reinforces that the virtualization system includes both the centralized control component and the underlying physical hardware resources as part of the same managed environment. A POSITA would have understood that this is a virtualization system in which a control computer manages one or more physical resources that host the virtualized assets.

[11.3] wherein each of the one or more physical resources execute at least two virtual assets that comprise software instantiations of computer systems and

237. The Hirschfeld-Smith-Vasilevsky System provides physical servers 108 (*physical resources*) that each execute multiple logical servers 106 (*at least two virtual assets*), where the logical servers are software-based virtualized computing instances and therefore are software instantiations of computer systems. MS-1006, 1:63-2:7, 6:34-46, 7:63-8:10, 10:9-31. Hirschfeld describes that virtualization technology enables multiple logical servers to operate on a single

physical computer and that multiple logical servers may be linked to a given physical server. *Id.*

238. As discussed for [1.2], Hirschfeld describes a virtualized logical server cloud that includes a bank of physical servers and a larger set of logical servers, with the number of logical servers capable of exceeding the number of physical servers. MS-1006, 6:28-63, 7:63-8:10. Hirschfeld also describes specific configurations in which “the logical servers LS 1-4 are linked to the PS A, the logical servers LS 5-8 are linked to the PS B, and the logical servers LS 9-N are linked to the PS C.” *Id.*, 7:63-8:10. A POSITA would have understood from those disclosed configurations that each illustrated physical server executes at least two logical servers because each identified physical server is shown supporting multiple separately identified logical servers. The logical servers each operate virtualized server instances running on the underlying hardware because Hirschfeld describes them as logical servers within a virtualization environment that share physical-server resources while remaining logically separate. *Id.*, 1:63-2:7, 10:9-31.

239. Hirschfeld further describes that the operating system of each logical server “is executed independently from every other LS OS 326, or otherwise gives every appearance of independent operation from the user’s perspective.” MS-1006, 10:9-31. A POSITA would have understood that description to mean that each

logical server is an example of a *software instantiation of a computer system*

because it has its own operating system environment, executes independently from other logical servers, and presents itself as an independent server environment to the user. This is the kind of virtualization-based software implementation that makes a logical server function as a computer system instance even though it shares underlying hardware with other logical servers on the same physical server.

Id., 1:63-2:7, 10:9-31.

240. Smith's disclosure is consistent with the understanding discussed above. MS-1010, 4-5, 11, 369. In Smith, a virtual machine is implemented by adding a layer of software to a real machine, that such virtualizing software is often referred to as a VMM, and that multiple virtual machines can be implemented on a single hardware platform. *Id.* A POSITA would have understood from Smith that virtualized server instances of the type contemplated by Hirschfeld are software-implemented computer-system instances running on shared hardware through virtualization software. A POSITA would also have understood that when Hirschfeld describes multiple logical servers operating independently on a single physical server, that configuration are examples of *multiple software instantiations of computer systems executing on the same physical resource in a virtualization environment*.

[11.4] the virtual assets are managed by virtualization layer instances running on each of the one or more physical resources;

241. The Hirschfeld-Smith-Vasilevsky System provides physical servers 108 (***one or more physical resources***) that each execute virtualization software 229 or a Smith-like VMM (***virtualization layer instances***) to manage the logical servers 106 (***virtual assets***) linked to and executing on those physical servers. MS-1006, 9:35-57; MS-1010, 369. Hirschfeld describes that each physical server executes virtualization software and that the virtualization software manages the physical resources of the logical servers linked thereto. MS-1006, 9:35-57.

242. As discussed for [1.3], Hirschfeld describes a configuration in which “[e]ach physical server 108 executes virtualization software 229,” and that virtualization software 229 “manages physical resources of the logical servers 106 linked thereto.” MS-1006, 9:35-57. A POSITA would have understood that virtualization software 229 is an example of a ***virtualization layer instance*** running on each physical server because Hirschfeld places that software on the physical server itself and assigns it the function of managing the resources used by the linked logical servers. The logical servers are managed by that software layer because Hirschfeld ties the management function directly to the logical servers linked to the particular physical server on which the virtualization software is running. *Id.* That is consistent with a virtualization architecture in which each host executes a local software layer that mediates and manages the operation of the

virtualized server instances running on that host.

243. Hirschfeld further describes that each logical server includes virtualized resources, including network, storage, memory, processor, and KVM resources, and that those logical-server resources are virtualized to the corresponding physical resources of the physical server. MS-1006, 8:66-9:20, 10:32-56. The virtualization software is a layer through which the logical servers obtain and use underlying host resources. This resource virtualization function is a management function of the virtualization layer because it is what allows multiple logical servers to operate as separate virtualized systems while sharing the same underlying physical hardware. *Id.* The described virtualization of logical-server resources to physical-server resources therefore supports the understanding that the virtual assets are managed by virtualization layer instances running on each physical resource.

244. Smith's disclosure is consistent with the understanding discussed above and helps explain how a POSITA would have implemented Hirschfeld's virtualization software. MS-1010, 4, 11, 369. Smith describes that a virtual machine is implemented by adding a layer of software to a real machine, that such virtualizing software is often referred to as a VMM, and that the VMM manages the allocation of, and access to, the hardware resources of the host platform. *Id.* A POSITA would have understood that a Smith-like VMM is a well-known example

of the type of virtualization layer Hirschfeld contemplates because it runs on the physical host and manages how multiple guest or logical systems use the host's hardware resources. A POSITA would also have understood that Hirschfeld's virtualization software 229 and Smith's VMM perform the same relevant role for this limitation: each is a software layer running on the physical resource that manages the virtualized server instances executing on that resource.

[11.5] requesting, by the user, provisioning of a virtual asset of the one or more virtual assets in the virtualization system; and

245. The Hirschfeld-Smith-Vasilevsky System receives, from a user interacting with the SCM 102 through a user interface 104 (***requesting, by the user***), a request to provision a selected logical server 106 (***virtual asset***) within the virtualized logical server cloud (***virtualization system***). MS-1006, 4:15-28, 6:28-7:30. Hirschfeld describes a user-facing SCM environment in which a remote user accesses the logical server cloud through the user interface and SCM and selects logical servers for use within that managed virtualized environment. *Id.*

246. As discussed for [1.4] and [1.5], Hirschfeld describes that a user of a remote terminal may log in through the user interface and SCM to access logical servers in the virtualized logical server cloud. MS-1006, 6:28-7:30. Hirschfeld also describes a system in which logical servers may be created, maintained, replicated, moved, or otherwise made available within the cloud under SCM control. *Id.*, 4:15-28, 7:24-30. A POSITA would have understood that when a user interacts with the

SCM to select and obtain a logical server for use, that interaction is an example of *a request for provisioning of a virtual asset within the virtualization system* because the SCM is the centralized management component that controls the availability and assignment of logical servers in the cloud. A POSITA would also have understood that, in a managed virtualization environment of this type, making a selected logical server available for the user necessarily involves management-side actions to instantiate, activate, allocate, or otherwise prepare that logical server on physical-server resources. *Id.*, 6:7-15, 7:24-30.

247. Hirschfeld's user-driven interaction with the SCM is a type of provisioning request because Hirschfeld separates logical-server access from fixed physical-server identity and instead places server availability under centralized SCM control. MS-1006, 2:50-59, 7:32-62, 17:9-24. That architecture means the user interacts with the management layer that determines which logical server is presented and how it is associated with underlying physical resources. *Id.* This kind of management-mediated selection and enablement of a logical server is functionally similar to provisioning in a virtualization context because the requested virtual asset is made available through centralized control, mapping, and resource allocation rather than through a static one-to-one server assignment. *Id.* Hirschfeld's disclosure that logical servers exist independently of specific physical servers and may be non-deterministically allocated to physical resources further

supports that understanding because it indicates that the system must perform management actions to provide the selected logical server instance in response to the user's interaction. *Id.*, 17:9-24.

248. Vasilevsky's disclosure is consistent with the understanding discussed above. MS-1005, [0074]-[0079]. Vasilevsky describes management functionality through which virtualized computing instances are controlled and resources are allocated or deallocated through an interface available to administrators or programs. *Id.* A POSITA would have understood from Vasilevsky that, in virtualized environments, provisioning of a virtualized computing instance is performed through management plane interactions responsive to a user or administrator request. A POSITA would also have understood that Hirschfeld's workflow fits comfortably within that same technical model, with the user's request to obtain a logical server corresponding to a request to provision a virtual asset in the virtualization system.

[11.6] wherein the provisioned virtual asset is a virtual storage and the virtual storage comprises one or more logical storage units mapped to one or more physical storage drives.

249. The Hirschfeld-Smith-Vasilevsky System provides a selected logical server 106 that, once provisioned in response to a user request (***provisioned virtual asset***), may be configured to function as a storage server for the network and thus as a ***virtual storage***. MS-1006, 8:66-9:20; MS-1005, [0011]. Hirschfeld describes

that storage resources may be divided into multiple physical and logical devices, including FTP and NAS, and that access to storage resources is preferably logical rather than physical so that they act like resource clouds. MS-1006, 8:66-9:20.

Vasilevsky describes that specialized storage hardware functions may be replaced by a “software-only application that is executed on a virtual server,” including “Network Attached Storage (NAS) functions.” MS-1005, [0011].

250. Hirschfeld’s storage resource disclosure provides the technical context for understanding how a provisioned logical server may function as a virtual storage. MS-1006, 8:66-9:20. Hirschfeld describes that each logical server includes a storage resource and that storage resources may be implemented using logical rather than physical access models so they behave as resource clouds. *Id.* A POSITA would have understood from that description that Hirschfeld’s logical servers are not limited to generic compute roles, but may be configured to provide storage-oriented services within the virtualized environment because Hirschfeld treats storage as one of the virtualized resources associated with the logical server. A POSITA would also have understood that a logical server configured to provide NAS-like or similar storage functionality is acting as an example of a *virtual storage* in the sense that the storage capability is presented through a software-defined, logical server instance. *Id.*

251. Vasilevsky’s disclosure reinforces the understanding discussed above

by describing a virtualized environment in which storage functionality is implemented on a virtual server rather than on specialized dedicated hardware. MS-1005, [0011], [0150], [0166]. Vasilevsky describes that NAS functions may be provided by a software-only application executing on a virtual server and also describes a mapping database that defines mappings between virtual resources and physical hardware. *Id.* A POSITA would have understood from those teachings that a provisioned virtualized server instance may serve as the virtual-storage component of the system when it is configured to provide storage services using underlying mapped storage resources. This is technically compatible with Hirschfeld's logical-server-cloud architecture because Hirschfeld already contemplates logical access to storage resources and virtualized storage-related server roles within the managed environment. MS-1006, 8:66-9:20; MS-1005, [0011].

5. *Claim 12*

[12] The method of claim 11 further comprising requesting, by a management application programming interface (API) of the virtualization system, the provisioning of the virtual storage.

252. The Hirschfeld-Smith-Vasilevsky System enables provisioning operations to be requested through a management interface that includes a programmatic interface (***application programming interface (API) of the virtualization system***). This includes provisioning operations directed to storage

resources allocated to a provisioned logical server or virtual server (*provisioning of the virtual storage*). MS-1005, [0014], [0074]-[0079]. Vasilevsky describes management-side control of resource allocation in a virtual computing system through interfaces that include programmatic access paths. *Id.*

253. Vasilevsky describes that control of allocating and deallocating resources to a virtual computing system may be provided “through an interface (e.g., via a CLI, or GUI) or to other programs (e.g., via a programmatic interface).” MS-1005, [0078]. This programmatic interface is an example of a *management API* because it is part of the virtualization system’s management plane and is used to control resource-allocation operations for virtualized computing instances. Provisioning is one type of such management operation because bringing a virtualized instance into service requires management-side allocation of the resources that instance will use. *Id.*, [0074]-[0079]. Vasilevsky’s description of management functionality available not only to administrators through UI-based interfaces but also to other programs through a programmatic interface further supports the understanding that the virtualization system includes an API-based mechanism for requesting provisioning operations. *Id.*, [0078].

254. The Hirschfeld-Smith-Vasilevsky System would have provided a similar type of provisioning of virtual storage. MS-1005, [0011], [0150], [0166]; MS-1006, 10:32-56. As discussed for [11.6], the combined system would have

provided a selected logical server that may function as a ***virtual storage*** and that utilizes virtualized storage constructs ***mapped*** to underlying physical storage devices. MS-1005, [0011], [0150], [0166]. A POSITA would have understood that when the system provisions such a storage-capable logical server, the associated storage allocation and mapping operations are part of the same management-controlled provisioning workflow addressed by Vasilevsky's programmatic interface. Requesting provisioning of the virtual storage through the API does not require a separate storage-only interface, because the management API governs allocation of virtualized resources generally, including storage-related resources assigned to the virtualized server instance. Id., [0074]-[0079], [0150].

6. *Claim 13*

[13] The method of claim 12 further comprising configuring, by the user, the virtual storage.

255. The Hirschfeld-Smith-Vasilevsky System enables a user to configure the ***virtual storage*** provided by a provisioned logical server that functions as a storage-oriented virtual asset. MS-1006, 8:66-9:20; MS-1005, [0074]-[0079], [0084], [0150]. Hirschfeld describes a virtualized environment in which storage resources are provided logically rather than physically, and Vasilevsky describes management functionality through which virtualized resources are allocated, associated, and controlled through user-facing and programmatic interfaces. MS-1006, 8:66-9:20; MS-1005, [0074]-[0079], [0084], [0150].

256. As discussed for [6] and [11.6], Hirschfeld describes logical servers that include storage resources and describes storage access as logical rather than physical so that the storage resources act like resource clouds. MS-1006, 8:66-9:20. Vasilevsky describes a management framework in which resources for virtual servers are configured through a management infrastructure that permits interaction by administrators, programs, and management-side components, and also maintains a mapping database defining mappings between virtual resources and physical hardware. MS-1005, [0074]-[0079], [0084], [0150]. Configuring the virtual storage in such a system includes selecting, allocating, associating, or otherwise defining the storage-related virtual resources that will be presented through the provisioned logical server. This is a type of user-side configuration because the management framework is disclosed as operating through interfaces by which a user or administrator requests and controls resource assignments for the virtualized instance. *Id.*, [0078], [0084].

257. A POSITA would have also understood that, in a virtualized storage context, “configuring” the virtual storage does not require low-level physical disk manipulation by the user. MS-1005, [0150], [0166]. The user-facing configuration occurs at the virtualized level by defining the storage-related virtual resources and the manner in which they are assigned to the virtual server, while the system maintains the corresponding mappings to underlying physical storage devices. *Id.*

That understanding is consistent with Hirschfeld's logical resource model because Hirschfeld describes storage resources as logically accessed cloud-like resources rather than as user-managed physical drives. MS-1006, 8:66-9:20. It is also consistent with Vasilevsky's management model because Vasilevsky describes virtual-resource allocation and control through management interfaces while separately maintaining the virtual-to-physical relationships in the mapping database. MS-1005, [0074]-[0079], [0150].

7. *Claim 14*

[14] The method of claim 13 further comprising receiving, by the user, an inventory of the virtual storage.

258. The Hirschfeld-Smith-Vasilevsky System enables a user, through a management plane interaction used for requesting provisioning, to receive a listing of available virtual storage resources (**inventory of the virtual storage**) presented for selection or assignment in connection with provisioning operations. MS-1006, 6:28-63, 7:32-48; MS-1005, [0077]-[0079], [0082]-[0084], [0150]. Hirschfeld describes a user-facing SCM pathway through which a user submits requests and receives system-provided information associated with accessing and managing logical servers, and Vasilevsky describes presentation of unassigned resources as available for assignment within the virtualization-management framework. MS-1006, 6:28-63, 7:32-48; MS-1005, [0082].

259. As discussed for [13], Hirschfeld describes that a user interacts with

the virtualization environment through the SCM and associated user interface, which provides the pathway for management-side interaction with logical servers in the cloud. MS-1006, 6:28-63, 7:32-48. A POSITA would have understood from that architecture that Hirschfeld supplies the user-facing mechanism through which information relevant to selection, access, and management of virtualized resources is communicated to the user because the SCM is the centralized point through which the user logs in, selects logical servers, and initiates management-related actions. *Id.* A POSITA would also have understood that, in a combined system using Vasilevsky's resource-assignment framework, the same management-side pathway would be used to present available storage-related virtual resources to the user in connection with provisioning because that is the interface through which the user interacts with the managed virtualized environment. *Id.*

260. Vasilevsky describes that unassigned resources may be “grouped into a ‘pool’ of unassigned resources and presented to an administrator or program as being available for assignment.” MS-1005, [0082]. A POSITA would have understood that this presentation of available resources is an example of an *inventory* because it is a system-provided listing or presentation of resources available to be selected and assigned within the virtualization system.

261. A POSITA would have also understood that the presented inventory includes storage-related virtual resources in the combined system. MS-1005,

[0072], [0150], [0166]. Vasilevsky includes storage among the resources allocated and deallocated to virtual servers and describes a mapping database that defines mappings of virtual resources to physical hardware, including physical storage devices such as disks. *Id.* That disclosure indicates that storage-related resources are part of the same managed resource universe from which unassigned resources are pooled and presented for assignment. A POSITA would also have understood that, when the user requests provisioning of virtual storage in the Hirschfeld-Smith-Vasilevsky System, the system would present available storage-related virtual resources, such as virtual storage entities or logical storage units available for assignment, through the management pathway provided by the SCM and user interface. MS-1006, 6:28-63, 7:32-48; MS-1005, [0082], [0150], [0166]. In that way, the user receives an inventory of the virtual storage to support the requested provisioning.

H. Ground 2D: Claims 9 And 10 Are Obvious Over The Combination of Hirschfeld, Smith, And Fujuno

1. Claim 9

[9] The method of claim 1, wherein providing the virtual assets further comprise providing a logical network resource.

262. The Hirschfeld-Smith-Fujino System would have provided a provisioned logical server (*virtual asset*) configured to utilize a software-defined representation of network connectivity. The SAN configuration information (*logical network resource*) specifies connectivity between host computers and

storage systems by identifying physical devices along the communication path, such as fiber channel switches and endpoint ports. MS-1006, 6:28-63, 10:32-56; MS-1008, 6:27-36, 7:35-43, 17:40-54.

263. As I discussed for [1.1]-[1.5], Hirschfeld describes a virtualization environment in which logical servers are provided on physical servers under control of the SCM and are accessed as virtualized server instances within the managed logical server cloud. MS-1006, 6:28-63, 10:32-56. Hirschfeld also describes that logical server resources include network resources virtualized relative to the underlying physical-server resources. *Id.*, 10:32-56. A POSITA would have understood from that architecture that providing the logical servers includes providing the network-side resources those logical servers use to communicate within the managed environment. This is because the logical servers are described as virtualized computing instances with their own virtualized resource sets. Network connectivity is part of the resource framework supporting those virtual assets because Hirschfeld treats network resources as among the resources virtualized for the logical servers so they can operate with virtual independence on shared infrastructure. *Id.*

264. In the Hirschfeld-Smith-Fujino System, Fujino provides an example of a technique for representing network connectivity as a ***logical network resource***. MS-1008, 5:31-45, 6:27-36, 7:35-43, 17:40-54. Fujino describes a SAN

architecture in which host-side computers communicate with storage systems through SAN infrastructure including fiber channel switches and ports. *Id.* The system maintains SAN configuration information identifying the physical devices along the communication path. *Id.* This SAN configuration information is a logical (or software-defined) representation of network connectivity because it captures the connectivity relationship in maintained information rather than requiring a user or administrator to work directly from the physical hardware layout itself. The represented communication path is still a network resource even though it is storage-oriented, because it defines how the provisioned logical server reaches storage over the networked SAN infrastructure. *Id.*

265. A POSITA would have further understood that this software-defined SAN connectivity is provided as part of providing the virtual asset in the combined system. MS-1006, 10:32-56; MS-1008, 6:27-36, 7:35-43. Hirschfeld's logical servers operate using virtualized resource sets (including network resources) while Fujino describes maintained SAN configuration information that specifies the underlying connectivity path used to reach storage resources. MS-1006, 10:32-56; MS-1008, 6:27-36, 7:35-43. In the combined system, the logical server's ability to use storage network connectivity would therefore be supported by the software-defined SAN connectivity information described by Fujino. This maintained connectivity information is an example of a logical network resource because it

provides the virtualized server environment with a defined network path in logical form while identifying the physical devices that implement that path.

2. *Claim 10*

[10] The method of claim 9, wherein providing the logical network resource further comprises mapping a logical network unit to a physical network device.

266. The Hirschfeld-Smith-Fujino System provides a software-defined network path representation, such as information maintained in the SAN configuration information and related communication-path information (**logical network unit**). This information is associated with the underlying network hardware (**physical network device**), such as physical ports of the storage control device and intervening fiber channel switch hardware used to implement the communication path. MS-1006, 15:10-17; MS-1008, 6:27-36, 6:40-49, 7:35-43, 27:27-41.

267. As discussed for [9], Fujino describes maintaining SAN configuration information that defines connectivity between endpoints by identifying the physical devices along the path. MS-1008, 6:27-36, 7:35-43. Fujino also describes that the SAN may be constituted by a plurality of fiber channel switches and that the storage control device includes one or more physical ports for connecting to the SAN. *Id.*, 6:27-49. A POSITA would have understood that the maintained path information is a logical network unit because it represents the communication path

in software or management information rather than as the bare physical hardware itself. This logical representation maps to physical network devices because the maintained information identifies the actual switches and ports that realize the represented path. *Id.*, 7:35-43.

268. Fujino further describes SAN configuration information that includes “an ID for the port of the storage control device 27, and one or more IDs corresponding respectively to the one or more FC-SWs existing between these ports.” MS-1008, 27:27-41. This is a mapping relationship because the logical path representation is defined by identifiers corresponding to the specific underlying physical devices used in that path. This is an association between the path as represented in the system’s management information and the particular physical network devices that implement it. The use of port IDs and FC-switch IDs ties the logical representation directly to the physical network hardware. *Id.* That is the type of relationship a POSITA would have recognized as mapping a logical network unit to a physical network device.

269. Hirschfeld is consistent with the understanding discussed above because it describes that, when logical servers are moved or reconfigured, “network hardware mappings (e.g., port mappings) are modified as necessary to maintain the logical net configuration.” MS-1006, 15:10-17. A POSITA would have understood from Hirschfeld that the virtualization environment already

contemplates a relationship between logical network configuration and underlying physical network hardware. Fujino also provides an example of an implementation of this relationship by representing the communication path in maintained SAN configuration information tied to identified ports and switches.

XI. CONCLUSION

270. For all the reasons I have noted in the foregoing paragraphs, claims 1-14 of the '915 patent are obvious in view of the references discussed above.

271. I currently hold the opinions set expressed in this declaration. But my analysis may continue, and I may acquire additional information and/or attain supplemental insights that may result in added observations.

APPENDIX A

Michael Mitzenmacher
michaelm@eecs.harvard.edu
33 Cary Avenue, Lexington MA 02421
617-496-7172

Research Interests	Design and Analysis of Algorithms; Machine Learning Systems; Networks and Data Transmission; Databases.
Education	UNIVERSITY OF CALIFORNIA AT BERKELEY, Berkeley, CA Ph.D. in Computer Science awarded December, 1996. Dissertation: The Power of Two Choices in Randomized Load Balancing. Advisor: Alistair Sinclair. GPA: 4.0/4.0 CAMBRIDGE UNIVERSITY, Cambridge, England Attended as one of ten recipients of the Churchill Fellowship. Cambridge C.A.S. in Mathematics with highest distinction awarded June 1992. HARVARD COLLEGE, Cambridge, MA B.A. in Mathematics with Computer Science, summa cum laude, awarded June 1991.
Employment	HARVARD UNIVERSITY, Cambridge, MA <i>Spring 1999-present</i> Assistant professor (from Jan. 1999 -July 2002), Associate professor (from July 2002-January 2005), Professor (from Jan. 2005-present), Area Dean for Computer Science (from July 2010-June 2013), Area Co-Chair for Computer Science (from July 2018-June 2019). Teach undergraduate courses including “Introduction to Algorithms and Data Structures” and “Discrete Mathematics for Computer Science”, and graduate courses covering topics in randomized algorithms, algorithms for networks, compression, coding, cryptography, and information retrieval. DIGITAL SYSTEMS RESEARCH CENTER, Palo Alto, CA <i>Fall 1996-Winter 1998</i> Research scientist. Projects included work on information retrieval on the Web, erasure codes, error-correcting codes, on-line algorithms, and load balancing. Co-inventor for twelve submitted patents. SANTA CLARA UNIVERSITY, Santa Clara, CA <i>Spring 1997</i> Guest professor for the undergraduate class “Introduction to Algorithms.” Consultant: I consult on intellectual property issues as an expert witness and in other capacities. As an expert witness, I have testified in multiple trials. I have also consulted for several technology companies and research laboratories, including Adverplex (Cogolabs), Akamai, AT&T, Digital Fountain, eHarmony, Fluent Mobile (Fiksu), Google, Huawei, ITA Software, JobSync, Microsoft, Mitsubishi Research Laboratories, Weta Digital, and Yahoo.
Funding	NSF CCF-2101140: AF: Small: Algorithms and Data Structures with Predictions. PI: Michael Mitzenmacher. Total grant: \$400,000. 7/21-6/24. NSF CNS-2107078: CNS Core: Medium: Approximation and Randomization in the Programmable Data Plane. co-PIs: Minlan Yu. Total grant: \$1,200,000. 7/21-6/25.

NSF DMS-2023528: Foundations of Data Science Institute (Harvard University). 9/20-8/25.

NSF CCF-1563710: CIF: NeTS: Medium: Collaborative Research: Unifying Data Synchronization. co-PIs: David Starobinksi, Ari Trachtenberg. Total grant: \$400,000. 7/16-6/20.

NSF CCF-1535705: AitF: FULL: Collaborative Research: Better Hashing for Applications: From Nuts & Bolts to Asymptotics. co-PIs: David Andersen. Total grant: \$250,000. 9/15-8/19.

NSF CCF-1320231: AF: Small: Data Synchronization : Theory, Algorithms, and Practice PI: Michael Mitzenmacher. Total grant: \$399,370. 9/13-8/16.

NSF CNS-1228598: TWC: Medium: Collaborative: Privacy-Preserving Distributed Storage and Computation. co-PIs: Michael Goodrich, Roberto Tamassia. Total grant: \$400,000. 8/12-8/16.

NSF IIS-0964473: HCC: Medium: Collaborative Research: Data-Parallel Hash Tables: Theory, Practice and Applications. co-PIs: Nina Amenta. Total grant: \$171,095. 8/10-7/13.

NSF CCF-0915922: AF : Small : The Theory and Practice of Hash-Based Algorithms and Data Structures. PI: Michael Mitzenmacher. Total grant: \$441,956. 8/09-7/12.

NSF CNS-0721491: NeTS FIND: A Network-Wide Hashing Infrastructure for Monitoring and Measurement. PI: Michael Mitzenmacher. Total grant: \$330,000. 9/07-8/10.

NSF CCF-0634923: Towards a Basic Understanding of Channels with Synchronization Errors. PI: Michael Mitzenmacher. Total grant: \$200,000. 9/06-8/09.

NSF CCR-0121154: ITR/SY Algorithmic Issues in Large Scale Dynamic Networks. co-PIs: Eli Upfal, Brown. Subcontract to Harvard. Total grant: \$502,000. 9/01-8/06.

NSF CCR-0118701: Low Density Parity Check Codes for Channels with Memory. co-PIs: Alek Kavcic. Total grant: \$510,000. 9/01-8/04.

NSF CCR-9983832: Dynamic Processes and Network Algorithms (CAREER). \$200,000. 7/00-6/04.

Dean's Competitive Fund for Promising Scholarship. 2018. \$49,333.

Google University Research Program. 8/13-8/14. PIs: Eddie Kohler and Michael Mitzenmacher \$56,000.

Google University Research Program. 12/11-12/12. \$25,000.

Yahoo! University Research Program. 7/11-6/12. \$10,000.

Google University Research Program. 12/09-12/10. \$60,000.

Yahoo! University Research Program. 9/09-8/10. \$10,000.

Google University Research Program. 8/08-7/09. \$75,000.

Yahoo! University Research Program. 9/07-8/08. \$25,000.

Yahoo! University Research Program. 9/06-8/07. \$50,000.

Cisco University Research Program. 8/08-7/09. \$80,000.

Cisco University Research Program. 8/07-7/08. \$83,000.

Cisco University Research Program. 8/06-7/07. \$80,000.

Cisco University Research Program. 8/05-7/06. \$72,000.

Alfred P. Sloan Research Fellowship. \$40,000. Awarded in 2000.

IBM Faculty Research Grant. \$10,000 Awarded in 2005.

IBM Faculty Research Grant. \$10,000 Awarded in 2003.

Mitsubishi Electronic Research Laboratory. \$10,000 for undergraduate research projects. Awarded in 2002.

Honors

Applied Cryptography and Network Security Conference Test of Time Paper Award (2025)

ACM Paris Kanellakis Theory and Practice Award (2020)

IEEE Fellow (2020)

IEEE Senior Member (2018)

ACM Fellow (2014)

ACM Symposium on Parallelism in Algorithms and Architectures Best Paper Award (2014)

World Wide Web Conference Best Paper Award (2014)

Royal Academy of Engineering Distinguished Visiting Fellowship (2010)

ACM SIGCOMM Test of Time Paper Award (2009)

IEEE Information Theory Society Best Paper Award (2002)

Alfred P. Sloan Research Fellowship (2000)

Sakrisson Award (for Ph.D. thesis at Berkeley) (1997)

NDSEG Graduate Fellowship (1992-95)

NSF Graduate Fellowship Winner (1991)

Churchill Fellowship (1991-92)

Hoopess Prize (for senior thesis at Harvard) (1991)

Phi Beta Kappa (1990)

Harvard Distinction in Teaching Award (1990)

Goldwater Fellowship (1989-91)

Professional Activities

Editorships and Boards:

Advisory Board, Hariri Institute at Boston University (2018-2021)

Editorial Board, Leibniz International Proceedings in Informatics (2009-2019)

Editorial Board, Communications of the ACM (2013-2018)

Science Board, Santa Fe Institute (2013-2019)

Guest Editor, SIAM Journal of Computing special issue for STOC 2009

SIAM Journal on Computing, Editor (2006-2010)

Internet Mathematics, Managing Editor

Guest Editor, Theory of Computing System special issue for SPAA 2002

Journal of Interconnection Networks, Editor

Organizing Committees:

Sublinear Algorithms Workshop Organizing Committee, FODSI (2022)
 Algorithms with Predictions Workshop Organizing Committee, STOC (2022)
 Machine Learning for Algorithms Workshop Organizing Committee, FODSI (2021)
 SIGACT Chair (2015-2018)
 Dagstuhl Seminar: Theory and Applications of Hashing (2017)
 EADS Summer School on Hashing: Theory and Applications (2014)
 ICERM Workshop on Stochastic Graph Models (2014)
 WAW 2013 Organizing Committee
 SIGACT Vice-Chair (2009-2012)
 General Chair, STOC (2010)
 Workshop on Randomized Algorithms and Random Graphs (2010)
 DARPA Information Science and Technology Study Groups (2007-2008)
 SIGACT Committee for the Advancement of Theoretical Computer Science (2005-2008)
 MSRI Workshop on Models of Real-World Random Networks (2005)
 FOCS 2003 Local Arrangements Chair
 Second Workshop on Randomized Algorithms and Random Graphs (2003)
 Workshop on Algorithms and Models for the Web Graph (2002)
 First Workshop on Randomized Algorithms and Random Graphs (2002)
 DIMACS Workshop on Quality of Service Issues in the Internet (2001)
 BU/NSF Workshop on Internet Measurement, Instrumentation, and Characterization (1999)
 FOCS 1998 Local Arrangements Chair

Program Committees:

ACM CoNEXT 2026 Program Committee
 ICLR 2026 Program Committee
 NSDI 2025 Program Committee
 HotNets 2024 Program Committee
 WADS 2023 Program Committee
 ACM CoNEXT 2023 Program Committee
 NeurIPS 2022 Program Committee
 ACM CoNEXT 2022 Program Committee
 ICML 2022 Program Committee
 SIGCOMM 2022 Program Committee
 NeurIPS 2021 Program Committee
 ICML 2021 Program Committee
 ACDA 2021 Program Committee
 ICLR 2021 Program Committee
 APoCS 2021 Program Committee
 ACM CoNEXT 2020 Program Committee
 NeurIPS 2020 Program Committee
 ICML 2020 Program Committee
 APoCS 2020 Program Committee
 NeurIPS 2019 Program Committee
 SIGCOMM 2019 Program Committee
 SOSA 2019 Program Committee (Co-Chair)
 ACM CoNEXT 2018 Program Committee
 ANALCO 2018 Program Committee
 SIGCOMM 2017 Program Committee
 NSDI 2017 Program Committee
 ACM CoNEXT 2016 Program Committee
 SIGCOMM 2016 Program Committee

ALLENEX 2016 Program Committee (Co-Chair)
ICALP 2016 Program Committee (Track C PC Chair)
ACM CoNEXT 2015 Program Committee
ICALP 2015 Program Committee
ITCS 2015 Program Committee
SIGCOMM 2014 Program Committee
ACM CoNEXT 2014 Program Committee
WSDM 2014 Senior Program Committee
WAW 2013 Program Committee (Co-Chair)
STOC 2013 Executive Committee
WSDM 2013 Senior Program Committee
SIGCOMM 2012 Program Committee
ISIT 2012 Program Committee
NSDI 2012 Program Committee
WAW 2011 Program Committee
ACM CoNEXT 2010 Program Committee
SIGCOMM 2010 Program Committee
NSDI 2010 Program Committee
FUN 2010 Program Committee
LATIN 2010 Program Committee
NetSciCom 2009 Program Committee
SIGCOMM 2009 Program Committee
STOC 2009 Program Committee (PC Chair)
NSDI 2009 Program Committee
ICALP 2008 Program Committee
SPAA 2008 Program Committee
WAW 2007 Program Committee
Analysis of Algorithms 2007 Program Committee
ANALCO 2007 Program Committee
ICALP 2007 Program Committee
RANDOM 2007 Program Committee
NCA 2006 Program Committee
STOC 2006 Program Committee
PODC 2005 Program Committee
ANALCO 2005 Program Committee
PODC 2004 Program Committee
SIGCOMM 2004 Program Committee
IPTPS 2004 Program Committee
Data Compression Conference 2004 Program Committee
ESA 2004 Program Committee
WAW 2003 Program Committee
FOCS 2003 Program Committee
Data Compression Conference 2003 Program Committee
ESA 2002 Program Committee
PODC 2002 Program Committee
SPAA 2002 Program Committee
STOC 2002 Program Committee
Data Compression Conference 2002 Program Committee
ALLENEX 2002 Program Committee
HiPC 2001 Program Committee
ISAAC 2000 Program Committee

RANDOM 2000 Program Committee
STACS 2000 Program Committee
FOCS 99 Program Committee
RANDOM 98 Program Committee

Reviewer for several conferences, journals, and grant panels.

Books

Probability and Computing: Randomized Algorithms and Probabilistic Analysis, by Michael Mitzenmacher and Eli Upfal. Published by Cambridge University Press in 2005. (2nd edition available in 2017.) This is a textbook meant for an advanced undergraduate or beginning graduate class. The textbook has been used in courses at Brown, Harvard, U. C. Berkeley, Univ. of Victoria, Tufts, Univ. of Mass. at Amherst, Purdue, U. Penn., and several other universities.

Conference and Journal Publications

“The Mixed Birth-death/death-Birth Moran Process,” with D. Brewster, Y. Huang, and M. Nowak. To appear in *Proceedings of 17th Innovations in Theoretical Computer Science Conference*, 2026.

“Spatiotemporal Sketch Disaggregation: Streaming Analytics with Heterogeneous Resources,” with J. Langlet, P. Chen, Z. Liu, R. Basat, and G. Antichi. To appear in *Proceedings of the 42nd IEEE International Conference on Data Engineering*, 2026.

“Better than Optimal: Improving Adaptive Stochastic Quantization Using Shared Randomness,” with R. Basat, Y. Ben-Itzhak, and S. Vargaftik. To appear in *Proceedings of the ACM on Measurement and Analysis of Computing Systems (SIGMETRICS)*, 2026.

“Call on MARS: Scheduling API-Augmented LLM Requests,” with R. Shahout, C. Liang, S. Xin, Q. Lao, Y. Cui, and M. Yu. To appear in *Proceedings of the 39th Conference on Neural Information Processing Systems*, 2025.

“HACK: Homomorphic Acceleration via Compression of the Key-Value Cache for Disaggregated LLM Inference,” with Z. Zhang, H. Shen, S. Vargaftik, R. Basat, and M. Yu. To appear in *Proceedings of ACM SIGCOMM*, 2025.

“Queueing, Predictions, and Large Language Models: Challenges and Open Problems,” with R. Shahout. To appear in *Stochastic Systems*.

“Don’t Stop Me Now: Embedding Based Scheduling For LLMs,” with R. Shahout, E. Malach, C. Liu, W. Jiang, and M. Yu. In *The Thirteenth International Conference on Learning Representations (ICLR)*, 2025.

“Parallel Peeling of Invertible Bloom Lookup Tables in a Constant Number of Rounds,” with M. Goodrich and R. Kitagawa. In *Proceedings of the 50th International Conference on Current Trends in Theory and Practice of Computer Science (SOFSEM)*, pp. 70-84, 2025.

“Efficient d -ary Cuckoo Hashing at High Load Factors by Bubbling Up,” with W. Kuszmaul. In *Proceedings of the ACM-SIAM Symposium on Discrete Algorithms*, pp. 3931-3952, 2025.

“Leveraging Parameterized Chernoff Bounds for Simplified Algorithm Analyses,” with M. Dillencourt and M. Goodrich. *Information Processing Letters*, 187:106516, 2025.

“Optimal and Approximate Adaptive Stochastic Quantization,” with R. Basat, Y. Ben-Itzhak, and S. Vargaftik. In *Proceedings of the 38th Conference on Neural Information Processing Systems*, 2024.

“SkipPredict: When to Invest in Predictions for Scheduling,” with R. Shahout. In *Proceedings of the 38th Conference on Neural Information Processing Systems*, 2024.

“Beyond Throughput and Compression Ratios: Towards High End-to-end Utility of Gradient Compression,” with W. Han, S. Vargaftik, B. Karp, and R. Basat. In *Proceedings of the Twenty-Third ACM Workshop on Hot Topics in Networks*, pp. 186-194, 2024.

“Learning-Augmented Frequency Estimation in Sliding Windows,” with R. Shahout and I. Sabek. In *Proceedings of the Workshop on Intelligent Classification of High-Speed Network Traffic (at ICNP ’24)*.

“Learning-Based Heavy Hitters and Flow Frequency Estimation in Streams,” with R. Shahout. In *Proceedings of the 32nd IEEE International Conference on Network Protocols*, 2024.

“Tabula: Efficiently Computing Nonlinear Activation Functions for Secure Neural Network Inference,” with M. Lam, V. J. Reddi, G. Wei, and D. Brooks. In *Transactions on Machine Learning Research*, 2024.

“Accelerating Federated Learning with Quick Distributed Mean Estimation,” with S. Vargaftik, R. Basat, A. Portnoy, G. Mendelson, and Y. Ben-Itzhak. In *Proceedings of the 41st International Conference on Machine Learning*, pp. 3410-3442, 2024.

“THC: Accelerating Distributed Deep Learning Using Tensor Homomorphic Compression,” with M. Li, R. Basat, S. Vargaftik, C. Lao, K. Xu, X. Tang, and, M. Yu. In *Proceedings of NSDI*, pp. 1191-1211, 2024.

“Separating k -Player from t -Player One-Way Communication, with Applications to Data Streams,” with E. Du, D. Woodruff, and G. Yang. In *Theory of Computing*, vol. 19, Article 10, pp. 1-44, 2023.

“Direct Telemetry Access,” with J. Langlet, R. Basat, G. Oliaro, M. Yu, and G. Antichi. In *Proceedings of ACM SIGCOMM*, pp. 832-849, 2023.

“Can Learned Models Replace Hash Functions?” with I. Sabek, K. Vaidya, D. Horn, A. Kipf, and T. Kraska. *Proceedings of the VLDB Endowment*, vol. 16, no. 3, pp. 532-545, 2022.

“Algorithmic Tools for Mining the Motif Structure of Networks,” with B. Matejek, T. Chen, and C. Tsourakakis. To appear in *Proceedings of the European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases*, 2022.

“Edge-colored directed subgraph enumeration on the connectome,” with B. Matejek, D. Wei, T. Chen, C. Tsourakakis, and H. Pfister. *Scientific Reports*, 2022.

“Algorithms with Predictions,” with S. Vassilvitskii. *Communications of the ACM*, Volume 65 Issue 7, pp. 33-35, 2022.

“EDEN: Communication-Efficient and Robust Distributed Mean Estimation for Federated Learning,” with S. Vargaftik, R. Basat, A. Portnoy, G. Mendelson, and Y. Ben-Itzhak. In

Proceedings of the 39th International Conference on Machine Learning, pp. 21984-22014, 2022.

“SNARF: A Learning-Enhanced Range Filter,” with K. Vaidya, S. Chatterjee, E. Knorr, S. Idreos, and T. Kraska. In *Proceedings of the VLDB Endowment*, 15(8):1632-1664, 2022.

“Proteus: A Self-Designing Range Filter,” with E. Knorr, B. Lemaire, A. Lim, S. Luo, H. Zhang, and S. Idreos. In *Proceedings of the SIGMOD International Conference on Management of Data*, pp. 1670-1684, 2022.

“Designing Sensitive Viral Diagnostics with Machine Learning,” with H. Metsky, N. Welch, P. Pillai, N. Haradhvala, L. Rumker, S. Mantena, Y. Zhang, D. Yang, C. Ackerman, J. Weller, P. Blainey, C. Myhrvold, and P. Sabeti. *Nature Biotechnology*, 2022.

“The Supermarket Model with Known and Predicted Service Times,” with M. Dell’Amico. *IEEE Transactions on Parallel and Distributed Systems*, 2022.

“Uniform Bounds for Scheduling with Job Size Estimates,” with Z. Scully and I. Grosof. In *Proceedings of 13th Innovations in Theoretical Computer Science Conference*, 114:1-114:30, 2022.

“DRIVE: One-bit Distributed Mean Estimation,” with S. Vargaftik, R. Basat, A. Portnoy, G. Mendelson, and Y. Ben-Itzhak. In *Proceedings of the 34th Conference on Neural Information Processing Systems*, pp. 362-377, 2021.

“Zero-CPU Collection with Direct Telemetry Access,” with J. Langlet, R. Basat, S.S. Ramanathan, G. Oliaro, M. Yu, and G. Antichi. In *Proceedings of the Twentieth ACM Workshop on Hot Topics in Networks*, pp. 108-115, 2021.

“Gradient Disaggregation: Breaking Privacy in Federated Learning by Reconstructing the User Participant Matrix,” with M. Lam, V. J. Reddi, D. Brooks, and G. Wei. In *Proceedings of the 38th International Conference on Machine Learning*, pp. 5959-5968, 2021.

“Putting the ‘Learning’ into Learning-Augmented Algorithms for Frequency Estimation,” with E. Du and F. Wang. In *Proceedings of the 38th International Conference on Machine Learning*, pp. 2860-2869, 2021.

“Queues with Small Advice.” In *Proceedings of the SIAM Conference on Applied and Computational Discrete Algorithms*, pp. 1-12, 2021.

“How to Send a Real Number Using a Single Bit (and Some Shared Randomness),” with R. Basat and S. Vargaftik. In *Proceedings of the 43rd International Colloquium on Automata, Languages, and Programming*, 25:1-25:20, 2021.

“SALSA: Self-Adjusting Lean Streaming Analytics,” with R. Basat, G. Einzinger, and S. Vargaftik. In *The 37th IEEE International Conference on Data Engineering*, pp. 864-875, 2021.

“Partitioned Learned Bloom Filters,” with K. Vaidya, E. Knorr, and T. Kraska. In *The Ninth International Conference on Learning Representations (ICLR)*, 2021.

“Improved Sublinear Time Algorithm for Longest Increasing Subsequence,” with S. Sedighin. In *Proceedings of the ACM-SIAM Symposium on Discrete Algorithms*, pp. 1934-1947, 2021.

“DISCOVering the heavy hitters with disaggregated sketches,” with V. Bruschi, R. Basat, Z. Liu, G. Antichi, and G. Bianchi. In *Proceedings of the 16th International Conference on Emerging Networks Experiments and Technologies (CoNEXT)*, pp. 536-537, 2020.

“Detecting Routing Loops in the Data Plane,” with J. Kucera, R. Basat, M. Kuka, G. Antichi, and M. Yu. In *Proceedings of the 16th International Conference on Emerging Networks Experiments and Technologies (CoNEXT)*, pp. 466-473, 2020.

“Optimal Learning of Joint Alignments with a Faulty Oracle,” with K. Larsen and C. Tsourakakis. In *Proceedings of ISIT*, pp.2492-2497, 2020.

“PINT: Probabilistic In-band Network Telemetry,” with R. Basat, S. Ramanathan, Y. Li, G. Antichi, and M. Yu. In *Proceedings of ACM SIGCOMM*, pp. 662-680, 2020.

“Dynamic Algorithms for LIS and Distance to Monotonicity,” with S. Seddighin. In *Proceedings of the 52nd Annual Symposium on Theory of Computing*, pp. 671-684, 2020.

“Clustering with a Faulty Oracle,” with K. Larsen and C. Tsourakakis. In *Proceedings of the Web Conference*, pp. 2831-2834, 2020.

“Prophets, Secretaries, and Maximizing the Probability of Choosing the Best,” with H. Esfandiari, M. Hajiaghayi, and B. Lucier. In *Proceedings of the 23rd International Conference on Artificial Intelligence and Statistics*, pp. 3717-3727, 2020.

“Faster and More Accurate Measurement through Additive-Error Counters,” with R. Basat, G. Einziger, and S. Vargaftik. In *Proceedings of IEEE INFOCOM*, pp. 1251-1260, 2020. o “Scheduling with Predictions and the Price of Misprediction.” In *Proceedings of 11th Innovations in Theoretical Computer Science Conference*, 14:1-14:18, 2020.

“Equitability, Interval Estimation, and Statistical Power,” with Y. Reshef, D. Reshef, and P. Sabeti. *Statistical Science*, (35)2:202-217, 2020.

“A Digital Fountain Retrospective,” with J. Byers and M. Luby. *ACM SIGCOMM Computer Communication Review*, 49(5), pp. 82-85, 2019.

“Robust Set Reconciliation via Locality Sensitive Hashing,” with T. Morgan. In *Proceedings of the 38th Symposium on Principles of Database Systems*, pp. 164-181, 2019.

“Online Pandora’s Boxes and Bandits,” with H. Esfandiari, M. Hajiaghayi, and B. Lucier. In *Proceedings of the 33rd AAAI Conference on Artificial Intelligence*, pp. 1885-1892, 2019.

“Arithmetic Progression Hypergraphs: Examining the Second Moment Method.” In *Proceedings of ANALCO*, pp. 127-134, 2019.

“A Bayesian Nonparametric View on Count-Min Sketch,” with R. Adams and D. Cai. In *Proceedings of the 31st Conference on Neural Information Processing Systems*, pp. 8782-8791, 2018.

“A Model for Learned Bloom Filters and Optimizing by Sandwiching.” In *Proceedings of the 31st Conference on Neural Information Processing Systems*, pp. 462-471, 2018.

“Metric Sublinear Algorithms via Linear Sampling,” with H. Esfandiari. In *Proceedings of the 59th Annual IEEE Symposium on Foundations of Computer Science*, pp. 11-22, 2018.

- “EMOMA: Exact Match in One Memory Access,” with S. Pontarelli and P. Reviriego. *IEEE Transactions on Knowledge and Data Engineering*, 30(11):2120-2133, 2018.
- “Weightless: Lossy Weight Encoding for Deep Neural Network Compression,” with B. Reagen, U. Gupta, R. Adolf, A. Rush, G. Wei, and D. Brooks. In *Proceedings of the 35th International Conference on Machine Learning*, pp. 4321-4330, 2018.
- “Load Thresholds for Cuckoo Hashing with Double Hashing,” with K. Panagiotou and S. Walzer. In *Proceedings of the 16th Scandinavian Symposium and Workshops on Algorithm Theory*, pp. 29:1-29:9, 2018.
- “Joint Alignment From Pairwise Differences with a Noisy Oracle,” with C. Tsourakakis. In *Proceedings of the 15th Workshop on Algorithms and Models for the Web Graph*, pp. 59-69, 2018. Journal version: *Internet Mathematics*, at <https://doi.org/10.24166/im.06.2019>, December 2020.
- “Reconciling Graphs and Sets of Sets,” with T. Morgan. In *Proceedings of the 37th Symposium on Principles of Database Systems*, pp. 33-47, 2018.
- “Simulated Annealing for JPEG Quantization,” with M. Hopkins and S. Wagner-Carena. In *Proceedings of the 2018 IEEE Data Compression Conference*, p. 414.
- “Adaptive Cuckoo Filters,” with S. Pontarelli and P. Reviriego. In *Proceedings of the ALENEX Conference*, pp. 36-47, 2018. Journal version: *ACM Journal of Experimental Algorithmics*, Article 1.1, March 2020.
- “An Empirical Comparison of the Maximal and Total Information Coefficients to Leading Measures of Dependence,” with D. Reshef, Y. Reshef, and P. Sabeti. *Annals of Applied Statistics*, 12(1):123-155, 2018.
- “Simple Multi-Party Set Reconciliation,” with R. Pagh. *Distributed Computing* 31(6):441-453, 2018.
- “Technical Perspective: Building a Better Hash Function.” *Communications of the ACM*, 60(7), p. 93, 2017.
- “Compresso: Efficient Compression of Segmentation Data For Connectomics,” with D. Haehn, F. Lekschas, B. Matejek, and H. Pfister. In *Proceedings of the 20th International Conference on Medical Image Computing and Computer Assisted Intervention*, pp. 781-788, 2017.
- “Auditable Data Structures,” with M. Goodrich, E. Kornaropoulos, and R. Tamassia. In *Proceedings of the European Symposium on Security and Privacy*, pp. 285-300, 2017.
- “2-3 Cuckoo Filters for Faster Triangle Listing and Set Intersection,” with D. Eppstein, M. Goodrich, and M. Torres. In *Proceedings of the 36th Symposium on Principles of Database Systems*, pp. 247-260, 2017.
- “Scalable motif-aware graph clustering,” with C. Tsourakakis and J. Pachocki. In *Proceedings of the 26th International Conference on the World Wide Web*, pp. 1451-1460, 2017.
- “Measuring Dependence Powerfully and Equitably,” with Y. Reshef, D. Reshef, H. Finucane, and P. Sabeti. *Journal of Machine Learning Research*, 17(212):1-63, 2016.

“Quantized Random Projections and Non-Linear Estimation of Cosine Similarity,” with P. Li and M. Slawski. In *Advances In Neural Information Processing Systems 29*, pp.2748-2756, 2016.

“Analyzing Distributed Join-Idle-Queue: A Fluid Limit Approach.” In *Proceedings of the 54th Annual Allerton Conference on Communication, Control, and Computing*, pp. 312-318, 2016.

“Hardness of Peeling with Stashes,” with V. Nathan. *Information Processing Letters*, 116(11), pp. 682-688, 2016.

“More Practical and Secure History-Independent Hash Tables,” with M. Goodrich, E. Kornaropoulos, and R. Tamassia. In *European Symposium on Research in Computer Security*, pp. 20-38, 2016.

“Models and Algorithms for Graph Watermarking,” with D. Eppstein, M. Goodrich, J. Lam, N. Mamano, and M. Torres. In *Proceedings of the Information Security Conference*, pp. 283-301, 2016.

“Better Bounds for Coalescing-Branching Random Walks on Graphs,” with R. Rajaraman and S. Roche. In *Proceedings of the 28th ACM Symposium on Parallel Algorithms and Architectures (SPAA)*, pp. 313-323, 2016. Journal version: *ACM Transactions on Parallel Computing*, 5(1), 2:1-2:23, 2018.

“OMASS: One Memory Access Set Separation,” with S. Pontarelli and P. Reviriego. *IEEE Transactions on Knowledge and Data Engineering*, 28(7):1940-1943, 2016.

“Voronoi Choice Games”, with M. Boppana, R. Hod, and T. Morgan. In *Proceedings of 43rd International Colloquium on Automata, Languages, and Programming*, 23:1-23:13, 2016.

“Space Lower Bounds for Itemset Frequency Sketches,” with E. Liberty, J. Thaler, and J. Ullman. In *Proceedings of the 35th Symposium on Principles of Database Systems*, pp. 441-454, 2016.

“Technical Perspective: Catching Lies (and Mistakes) in Offloaded Computation,” with J. Thaler. *Communications of the ACM*, 59(2), p. 102, 2016.

“A New Approach to Analyzing Robin Hood Hashing.” In *Proceedings of ANALCO*, pp. 10-24, 2016.

“More Analysis of Double Hashing for Balanced Allocations.” In *Proceedings of ANALCO*, pp. 1-9, 2016.

“Theory Without Experiments: Have We Gone Too Far?” *Communications of the ACM*, 58(9), pp. 40-42, 2015.

“Scaling Up Clustered Network Appliances with XBricks,” with D. Zhou, B. Fan, H. Lim, R. Wang, M. Kaminsky, D. Andersen, and A. Singh. In *Proceedings of ACM SIGCOMM*, 2015.

“Scalable Large Near-Clique Detection in Large-Scale Networks via Sampling,” with J. Pachocki, R. Peng, C. Tsourakakis, and S. Chen Xu. In *Proceedings of the 21st ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2015.

- “Repeated Deletion Channels,” with B. Haeupler. In *Proceedings of the IEEE Information Theory Workshop*, pp. 152-156, 2014.
- “Multi-Party Set Reconciliation Using Characteristic Polynomials,” with A. Boral. In *Proceedings of the 52nd Annual Allerton Conference on Communication, Control, and Computing*, pp. 1182-1187, 2014.
- “Cleaning Up the Record on the Maximal Information Coefficient and Equitability,” with D. Reshef, Y. Reshef, and P. Sabeti. *Proceedings of the National Academy of Sciences* 111(33):E3362-3, 2014.
- “Cuckoo Filter: Practically Better than Bloom,” with B. Fan, D.G. Andersen, and M. Kaminsky. In *Proceedings of the 10th International Conference on Emerging Networks Experiments and Technologies (CoNEXT)*, pp. 75-88, 2014.
- “Coding for Random Projections,” with P. Li and A. Shrivastava. In *Proceedings of the 31st International Conference on Machine Learning (ICML)*, pp. 676-684, 2014.
- “Balanced Allocations and Double Hashing.” In *Proceedings of the 26th ACM Symposium on Parallel Algorithms and Architectures (SPAA)*, pp. 331-342, 2014.
- “Parallel Peeling Algorithms,” with J. Jiang and J. Thaler. In *Proceedings of the 26th ACM Symposium on Parallel Algorithms and Architectures (SPAA)*, pp. 319-330, 2014. Journal version: *ACM Transactions on Parallel Computing*, 3(1), 7, 2016.
- “Wear Minimization for Cuckoo Hashing: How Not to Throw a Lot of Eggs into One Basket”, with D. Eppstein, M. Goodrich, and P. Pszona. In *Proceedings of the 13th International Symposium on Experimental Algorithms*, pp. 162-173, 2014.
- “Efficient Estimation for High Similarities using Odd Sketches,” with R. Pagh and N. Pham. In *Proceedings of the 23rd International Conference on the World Wide Web*, pp. 109-118, 2014.
- “Improving the Performance of Invertible Bloom Lookup Tables,” with S. Pontarelli and P. Reviriego. *Information Processing Letters*, 114(4), pp. 185-191, 2014.
- “Special Section on the Forty-First Annual ACM Symposium on Theory of Computing (STOC 2009),” with N. Immerlica, J. Katz, R. Servedio, and C. Umans. *SIAM Journal on Computing*, 41(6):1591-1592, 2012.
- “The Daily Deals Marketplace: Empirical Observations and Managerial Implications,” with J. Byers and G. Zervas. *AC SIGecom Exchanges*, 11(2):29-31, 2012.
- “Peeling Arguments and Double Hashing,” with J. Thaler. In *Proceedings of the 50th Annual Allerton Conference on Communication, Control, and Computing*, pp. 1118-1125, 2012.
- “The Complexity of Object Reconciliation, and Open Problems Related to Set Difference and Coding,” with G. Varghese. In *Proceedings of the 50th Annual Allerton Conference on Communication, Control, and Computing*, pp. 1126-1132, 2012.
- “Cache-Oblivious Dictionaries and Multimaps with Negligible Failure Probability,” with M. Goodrich, D. Hirschberg, and J. Thaler. In *Proceedings of the First Mediterranean Conference on Algorithms*, pp. 203-218, 2012.

“An Economic Analysis of User-Privacy Options in Ad-Supported Services,” with J. Feigenbaum and G. Zervas. In *Proceedings of the 8th Workshop on Internet and Network Economics*, pp. 30-43, 2012.

“Verifiable Computation with Massively Parallel Interactive Proofs,” with H. Pfister, M. Roberts, and J. Thaler. In *Proceedings of the USENIX HotCloud Workshop, 2012*.

“Biff (Bloom Filter) Codes: Fast Error Correction for Large Data Sets,” with G. Varghese. In *Proceedings of the International Symposium on Information Theory*, pp. 483-487, 2012.

“Continuous Time Channels with Interference,” with I. Ivan, J. Thaler, and H. Yuen. In *Proceedings of the International Symposium on Information Theory*, pp. 860-864, 2012.

“The Groupon Effect on Yelp Ratings: a Root Cause Analysis,” with J. Byers and G. Zervas. In *Proceedings of the ACM Conference on Electronic Commerce*, pp. 248-265, 2012.

“Anonymous Card Shuffling and Its Applications to Parallel Mixnets,” with M. Goodrich. In *Proceedings of 39th International Colloquium on Automata, Languages, and Programming*, pp. 549-560, 2012.

“Chernoff-Hoeffding Bounds for Markov Chains: Generalized and Simplified,” with K. Chung, H. Lam, and Z. Liu. In *Proceedings of the 29th International Symposium on Theoretical Aspects of Computer Science*, pp. 124-135, 2012.

“Practical oblivious storage,” with M. Goodrich, O. Ohrimenko and R. Tamassia. In *Proceedings of the Second ACM Conference on Data and Application Security and Privacy*, pp. 13-24, 2012.

“Hierarchical Heavy Hitters with the Space Saving Algorithm,” T. Steinke and J. Thaler. In *Proceedings of 2012 Meeting on Algorithm Engineering and Experiments*, pp. 160-174.

“Daily Deals: Prediction, Social Diffusion, and Reputational Ramifications,” with J. Byers and G. Zervas. In *Proceedings of the 5th International Conference on Web Search and Web Data Mining*, pp. 543-552, 2012.

“Practical Verified Computation with Streaming Interactive Proofs,” with G. Cormode and J. Thaler. In *Proceedings of Innovations in Theoretical Computer Science*, pp. 90-112, 2012.

“Information Dissemination via Random Walks in d -Dimensional Space,” with H. Lam, Z. Liu, X. Sun, and Y. Wang. In *Proceedings of the 23rd Annual ACM-SIAM Symposium on Discrete Algorithms*, pp. 1612-1622, 2012.

“Privacy-Preserving Group Data Access via Stateless Oblivious RAM Simulation,” with M. Goodrich and R. Tamassia. In *Proceedings of the 23rd Annual ACM-SIAM Symposium on Discrete Algorithms*, pp. 157-167, 2012.

“An Efficient Rigorous Approach for Identifying Statistically Significant Frequent Itemsets,” with A. Kirsch, A. Pietracaprina, G. Pucci, E. Upfal, and F. Vandin. *Journal of the ACM*, 59(3), 2012.

“Detecting Novel Associations in Large Datasets,” with D. Reshef, Y. Reshef, H. Finucane, S. Grossman, G. McVean, P. Turnbaugh, E. Lander, and P. Sabeti. *Science*, Vol. 3004, No. 6602, pp. 1518-1524, December 2011.

- “External-Memory Multimaps,” with E. Angelino, M. Goodrich, and J. Thaler. In *Proceedings of the 22nd International Symposium on Algorithms and Computation*, pp. 384-394, 2011. Journal version: *Algorithmica*, 67(1), pp. 23-48, 2013.
- “Brief announcement: large-scale multimaps,” with M. Goodrich. In *Proceedings of the 23rd ACM Symposium on Parallel Algorithms and Architectures (SPAA)*, pp. 259-260, 2011.
- “Oblivious RAM Simulation with Efficient Worst-Case Access Overhead,” with M. Goodrich, O. Ohrimenko, and R. Tamassia. In *Proceedings of 3rd ACM Cloud Computing Workshop*, pp. 95-100.
- “Invertible Bloom Lookup Tables,” with M. Goodrich. In *Proceedings of the 49th Annual Allerton Conference on Communication, Control, and Computing*, pp. 792-799, 2011.
- “Cuckoo Hashing with Pages,” with M. Dietzfelbinger and M. Rink. In *Proceedings of the 19th Annual European Symposium on Algorithms*, pp. 615-627, 2011.
- “Network Coding Meets TCP: Theory and Implementation,” with J.K. Sundararajan, D. Shah, M. Medard, S. Jakubczak, and J. Barros. *Proceedings of the IEEE*, 99(3), pp. 490-510, 2011.
- “Privacy-Preserving Access of Outsourced Data via Oblivious RAM Simulation,” with M. Goodrich. In *Proceedings of 38th International Colloquium on Automata, Languages, and Programming*, pp. 576-587, 2011.
- “Graption: A Graph-based P2P Traffic Classification Framework for the Internet Backbone,” with M. Iliofotou, H. Kim, M. Faloutsos, P. Pappu, and G. Varghese. *Computer Networks*, 55(8), pp. 1909-1920, 2011.
- “On the Zero-Error Capacity Threshold for Deletion Channels,” with I. Kash, J. Thaler, J. Ullman. In *Proceedings of the Information Theory and Applications Workshop*, pp. 1-5, 2011.
- “Heapable Sequences and Subsequences,” with J. Byers, B. Heeringa, and G. Zervas. In *Proceedings of ANALCO*, pp. 33-44, 2011.
- “An Introduction to Human Guided Search.” *ACM Crossroads*, 17(2):34-35, 2010.
- “Popularity is Everything: A New Approach to Protecting Passwords from Statistical Guessing Attacks,” with S. Schechter and C. Herley. In *Proceedings of the 5th Usenix Workshop on Hot Topics in Security*, pp. 1-8, 2010.
- “Streaming Graph Computations with a Helpful Advisor,” with G. Cormode and J. Thaler. In *Proceedings of the European Symposium on Algorithms*, pp. 231-242, 2010. Journal version: *Algorithmica*, 65(2), pp. 409-442, 2013.
- “Tight Thresholds for Cuckoo Hashing via XORSAT,” with M. Dietzfelbinger, A. Goerdt, A. Montanari, R. Pagh, and M. Rink. In *Proceedings of 37th International Colloquium on Automata, Languages, and Programming*, pp. 213-225, 2010.
- “Local cluster aggregation models of explosive percolation,” with R.M. D’Souza. *Physical Review Letters*, 14(19), p. 104-107, 2010.

“Tight Asymptotic Bounds for the Deletion Channel with Small Deletion Probabilities,” with A. Kalai and M. Sudan. In *Proceedings of the International Symposium on Information Theory*, pp. 997-1001, 2010.

“An Improved Analysis of the Lossy Difference Aggregator,” with H. Finucane. *Computer Communication Review*, 40(2), pp. 4-11, 2010.

“Information Asymmetries in Pay-Per-Bid Auctions: How Swoopo Makes Bank,” with J. Byers and G. Zervas. In *Proceedings of the ACM Conference on Electronic Commerce*, pp. 1-12, 2010.

“Carousel: Scalable Logging for Intrusion Prevention Systems”, with T. Lam and G. Varghese. In *Proceedings of NSDI 2010*, pp. 361-376.

“AMS Without 4-Wise Independence on Product Domains,” with V. Braverman, K. Chung, Z. Liu, and R. Ostrovsky. In *Proceedings of the 27th International Symposium on Theoretical Aspects of Computer Science*, pp. 119-130, 2010.

“Adaptive Weighing Designs for Keyword Value Computation,” with J. Byers and G. Zervas. In *Proceedings of the Third International Conference on Web Search and Web Data Mining* pp. 331-340, 2010.

“Human-guided search,” with G. Klau, N. Lesh, and J. Marks. *Journal of Heuristics*, 16(3), pp. 289-310, 2010.

“Real-Time Parallel Hashing on the GPU,” with D. Alcantara, A. Sharf, F. Abbasinejad, S. Sengupta, J. Owens, and N. Amenta. *ACM Transactions on Graphics*, 28(5), p. 154, 2009.

“Exploiting Dynamicity in Graph-based Traffic Analysis: Techniques and Applications,” with M. Iliofotou and M. Faloutsos. In *Proceedings of ACM Co-NEXT*, pp. 241-252, 2009.

“An Analysis of Random-Walk Cuckoo Hashing,” with A. Frieze and P. Mellsted. In *Proceedings of 2009 APPROX-RANDOM*, pp. 490-503, 2009. Journal version: *SIAM Journal on Computing*, 40(2), pp. 291-308, 2011.

“Some Open Questions Related to Cuckoo Hashing.” In *Proceedings of the 17th Annual European Symposium on Algorithms*, pp. 1-10, 2009.

“On Compressing Social Networks,” with F. Chierichetti, R. Kumar, S. Lattanzi, A. Panconesi, and P. Raghavan. In *Proceedings of the 15th ACM SIGKDD Int’l Conference on Knowledge Discovery and Data Mining*, pp. 219-228, 2009.

“An Efficient Rigorous Approach for Identifying Statistically Significant Frequent Itemsets,” with A. Kirsch, A. Pietracaprina, G. Pucci, E. Upfal, and F. Vandin. In *Proceedings of the 28th Symposium on Principles of Database Systems*, pp. 117-126, 2009.

“Using the Power of Two Choices to Improve Bloom Filters,” with Steven Lumetta. *Internet Mathematics*, vol. 4. no. 1, pp. 17-33, 2009.

“Network coding meets TCP,” with J. Sundararajan, D. Shah, M. Medard, and J. Barros. In *Proceedings of IEEE INFOCOM*, pp. 280-288, 2009.

- “An Economically Principled Generative Model of AS Graph Connectivity,” with J. Corbo, S. Jain, and D. Parkes. In *Proceedings of IEEE INFOCOM (Mini-conference)*, pp. 2941-2945, 2009.
- “Designing floating codes for expected performance,” with H. Finucane and Z. Liu. In *Proceedings of the 46th Annual Allerton Conference on Communication, Control, and Computing*, 2008. Journal version (with F. Chierichetti, H. Finucane, and Z. Liu): *IEEE Transactions on Information Theory*, vol 56, Issue 3, pp. 968-978, 2010.
- “On the performance of multiple choice hash tables with moves on deletions and inserts,” with A. Kirsch. In *Proceedings of the 46th Annual Allerton Conference on Communication, Control, and Computing*, pp. 1284-1290, 2008.
- “More Robust Hashing: Cuckoo Hashing with a Stash,” with A. Kirsch and U. Wieder. In *Proceedings of the 16th Annual European Symposium on Algorithms*, pp. 611-622, 2008. Journal version: *SIAM Journal on Computing*, 39(4), pp. 1543-1561, 2009.
- “A survey of results for deletion channels and related synchronization channels.” In *Proceedings of the 2008 Scandinavian Workshop on Algorithm Theory*, pp. 1-3, 2008. Journal version: “New Results and Open Problem for Channels with Synchronization.” *Probability Surveys*, 2009.
- “The Power of One Move: Hashing Schemes for Hardware,” with A. Kirsch. In *Proceedings of IEEE INFOCOM*, pp. 106-110, 2008. Journal version: *IEEE/ACM Transactions on Networking*, 18(6):1752-1765, 2010.
- “Distributed Beamforming with Binary Signaling,” with M. Johnson and K. Ramchandran. In *Proceedings of the 2008 Int’l Symposium on Information Theory (ISIT)*, pp. 890-894, 2008.
- “The Hiring Problem and Lake Wobegon Strategies,” with A. Broder, A. Kirsch, R. Kumar, E. Upfal, and S. Vassilvitskii. In *Proceedings of the 14th Annual ACM-SIAM Symposium on Discrete Algorithms*, pp. 1184-1193, 2008. Journal version: *SIAM Journal on Computing*, 39(4), pp. 1233-1255, 2009.
- “Why Simple Hash Functions Work: Exploiting the Entropy in a Data Stream,” with S. Vadhan. In *Proceedings of the 14th Annual ACM-SIAM Symposium on Discrete Algorithms*, pp. 746-755, 2008. Journal version: with K. Chung and S. Vadhan. *Theory of Computing*, 9, pp. 897-945-1255, 2013.
- “Trace reconstruction with constant deletion probability and related results,” with T. Holenstein, R. Panigrahy, and U. Wieder. In *Proceedings of the 14th Annual ACM-SIAM Symposium on Discrete Algorithms*, pp. 389-398, 2008.
- “Capacity bounds for sticky channels.” *IEEE Transactions on Information Theory*, vol 54, Issue 1, pp. 72-77, 2008.
- “Network Monitoring Using Traffic Dispersion Graphs,” with M. Iliofotou, P. Pappu, G. Varghese, M. Faloutsos, and S. Singh. In *Proceedings of the 7th ACM SIGCOMM Conference on Internet Measurement*, pp. 315-320, 2007.
- “HEXA: Compact Data Structures for Faster Packet Processing,” with S. Kumar, J. Turner, and P. Crowley. In *Proceedings of the IEEE International Conference on Network Protocols*, pp. 246-255, 2007.

- “Using a Queue to De-amortize Cuckoo Hashing in Hardware,” with A. Kirsch and M. Mitzenmacher. In *Proceedings of the 45th Annual Allerton Conference on Communication, Control, and Computing*, 2007.
- “Improved lower bounds for the capacity of i.i.d. deletion and duplication channels, with E. Drinea. *IEEE Transactions on Information Theory*, vol 53, Issue 8, pp. 2693-2714, 2007.
- “Wired Geometric Routing,” with J. Ledlie, P. Pietzuch, and M. Seltzer. In *Proceedings of the 6th International Workshop on Peer-to-Peer Systems (IPTPS)*, 2007.
- “Capacity Upper Bounds for Deletion Channels,” with S. Diggavi and H. Pfister. In *Proceedings of the 2007 Int’l Symposium on Information Theory (ISIT)*, pp. 1716-1720, 2007.
- “Codes for Deletion and Insertion Channels with Segmented Errors,” with Z. Liu. In *Proceedings of the 2007 Int’l Symposium on Information Theory (ISIT)*, pp. 846-850, 2007. Journal version: *IEEE Transactions on Information Theory*, vol 56, Issue 1, pp. 224-232, 2010.
- “Towards a theory of networked computation : Executive Summary,” with J. Feigenbaum. *ACM SIGACT News*, (4):22-26, 2006.
- “Bloom Filters via d-Left Hashing and Dynamic Bit Reassignment,” with F. Bonomi, R. Panigrahy, S. Singh, and G. Varghese. In *Proceedings of the Allerton Conference*, pp. 877-883, 2006.
- “Beyond Bloom Filters: From Approximate Membership Checks to Approximate State Machines,” with F. Bonomi, R. Panigrahy, S. Singh, and G. Varghese. In *Proceedings of ACM SIGCOMM*, pp. 315-326, 2006.
- “An Improved Construction for Counting Bloom Filters,” with F. Bonomi, R. Panigrahy, S. Singh, and G. Varghese. In *Proceedings of the European Symposium on Algorithms*, pp. 684-695, 2006.
- “Less Hashing, Same Performance: Building A Better Bloom Filter,” with A. Kirsch. In *Proceedings of the European Symposium on Algorithms*, pp. 456-467, 2006. Journal version: *Random Structures and Algorithms*, 33(2), pp. 187-218, 2008.
- “Stochastic Shortest Paths via Quasi-Convex Maximization,” with M. Brand, J. Kelner, and E. Nikolova. In *Proceedings of the European Symposium on Algorithms*, pp. 552-563, 2006.
- “Polynomial Time Low-Density Parity-Check Codes with Rates Very Close to the Capacity of the q -ary Random Deletion Channel for Large q .” *IEEE Transactions on Information Theory*, vol 52, Issue 12, pp. 5496-5501, 2006.
- “A Simple Lower Bound for the Capacity of the Deletion Channel,” with E. Drinea. *IEEE Transactions on Information Theory*, vol 52, Issue 10, pp. 4657-4660, 2006.
- “On Lower Bounds for the Capacity of Deletion Channels,” with E. Drinea. *IEEE Transactions on Information Theory*, vol 52, Issue 10, pp. 4648-4657, 2006.
- “On the Theory and Practice of Data Recovery with Multiple Versions.” In *Proceedings of the International Symposium on Information Theory*, pp. 982-986, 2006.

- “Fine-Grained Layered Multicast with STAIR,” with J. Byers, Gu-In Kwon, and M. Luby. *IEEE/ACM Transactions on Networking*, 14:1, pp. 81-93, 2006.
- “Distance-Sensitive Bloom Filters,” with A. Kirsch. In *Proceedings of the ALENEX Conference*, pp. 41-50, 2006.
- “BubbleSearch: A Simple Heuristic for Improving Priority-Based Greedy Algorithms,” with N. Lesh. *Information Processing Letters*, 97(4), pp. 161-169, 2006.
- “Network-Aware Overlays with Network Coordinates,” with P. Pietzuch, J. Ledlie, and M. Seltzer. In *Proceedings of the International Workshop on Dynamic Distributed Systems*, 2006.
- “Editorial: The Future of Power Law Research.” *Internet Mathematics*, vol. 2. no. 4, pp. 525-534, 2006.
- “X-Tolerant Test Response Compaction,” with S. Mitra, S. Lumetta, and N. Patil. *IEEE Design and Test of Computers*, vol. 22, no. 6, pp. 566-574, Nov/Dec 2005.
- “Privacy preserving keyword searches on remote encrypted data,” with Y.-C. Chang. In *Proceedings of the Third Intl. Conference on Applied Cryptography and Network Security*, pp. 442-455, 2005.
- “Simple Summaries for Multiple Choice Hashing,” with A. Kirsch. In *Proceedings of the 43rd Annual Allerton Conference on Communication, Control, and Computing*, 2005. Journal version: *IEEE/ACM Transactions on Networking*: 16(1), pp. 218-231, 2008.
- “A case study in large-scale interactive optimization,” with M. Chimani, N. Lesh, and C. Sidner. In *Proceedings of Artificial Intelligence and Applications 2005*, pp. 24-29.
- “New heuristic and interactive approaches to 2D rectangular strip packing,” with N. Lesh, A. McMahon, and J. Marks. *ACM Journal of Experimental Algorithms*, 10, 2005.
- “Multidimensional Balanced Allocations,” with A. Broder. In *Proceedings of the 11th Annual ACM-SIAM Symposium on Discrete Algorithms*, pp. 195-196, 2005.
- “Digital Fountains: A Survey and Look Forward.” 2004 Information Theory Workshop.
- “Improved Lower Bounds on the Capacity of I.I.D. Deletion Channels,” with E. Drinea. In *Proceedings of the 42nd Annual Allerton Conference on Communication, Control, and Computing*, 2004.
- “Improved Lower Bounds on the Capacity of Deletion Channels,” with E. Drinea. In *Proceedings of the 2004 IEEE International Symposium on Information Theory*.
- “Geometric Generalizations of the Power of Two Choices,” with J. Byers and J. Conside. In *Proceedings of the 16th ACM Symposium on Parallel Algorithms and Architectures (SPAA)*, pp. 54-63, 2004.
- “Interactive Data Summarization: An Example Application,” with N. Lesh. In *Proceedings of the 2004 Workshop on Advanced Visual Interfaces*, pp. 183-187.
- “On Power Law Distributions for Monkeys Typing Randomly: The Case of Unequal Probabilities,” with B. Conrad. *IEEE Transactions on Information Theory*, 50:7, pp. 1403-1414, 2004.

- “Dynamic Models for File Sizes and Double Pareto Distributions.” *Internet Mathematics*, 1(3):305-334, 2003.
- “A Scaling Result for Explosive Processes,” with R. Oliveira and J. Spencer. *Electronic Journal of Combinatorics*, 11(1), R31, 2004.
- “An Exhaustive Approach to 2D Strip Packing,” with N. Lesh, A. McMahon, and J. Marks. *Information Processing Letters*, 90(1), pp. 7-14, 2004.
- “New Models and Methods for File Size Distributions,” with B. Tworetzky. In *Proceedings of the 41st Annual Allerton Conference on Communication, Control, and Computing*, pp. 603-612, 2003.
- “Capacity Approaching Signal Constellations for Channels with Memory,” with A. Kavčić, and X. Ma. *IEEE Transactions on Information Theory*, 49:7, pp. 1636-1652, 2003.
- “The Power Spectra of Good Codes for Partial Response Channels,” with A. Kavcic and X. Ma. In *Proceedings of the 2003 IEEE International Symposium on Information Theory*, p. 45.
- “Concatenated Codes for Deletion Channels,” with J. Chen, C. Ng, and N. Varnica. In *Proceedings of the 2003 IEEE International Symposium on Information Theory*, p. 218.
- “Verification Codes for Deletion Channels.” In *Proceedings of the 2003 IEEE International Symposium on Information Theory*, p. 217.
- “Estimating and Comparing Entropies Across Written Natural Languages Using PPM Compression,” with F. Behr, V. Fossum, and D. Xiao. In *Proceedings of the 2003 IEEE Data Compression Conference*, p. 416.
- “Simple Load Balancing for Distributed Hash Tables,” with J. Byers, and J. Considine. Appears in the 2nd International Workshop on Peer-to-Peer Systems (IPTPS), pp. 80-87, 2003.
- “A Complete and Effective Move Set for Simplified Protein Folding,” with N. Lesh, and S. Whitesides. In *Proceedings of the 7th Annual International Conference on Research in Computational Molecular Biology*, pp 188-195, 2003.
- “Network Applications of Bloom Filters: A Survey,” with A. Broder. In *Proceedings of the 40th Annual Allerton Conference on Communication, Control, and Computing*, pp. 636-646, 2002. Journal version: *Internet Mathematics*, 1(4):485-509, 2003.
- “Verification Codes,” with M. Luby. In *Proceedings of the 40th Annual Allerton Conference on Communication, Control, and Computing*, pp. 38-47, 2002. Journal version: *IEEE Transactions on Information Theory*: 51(1), pp. 120-127, 2005.
- “Balls and Bins with Memory,” with B. Prabhakar and D. Shah. In *Proceedings of the 43rd Annual IEEE Symposium on Foundations of Computer Science*, pp. 799-808, 2002.
- “Informed Content Delivery Across Adaptive Overlay Networks,” with J. Byers, J. Considine, and S. Rost. In *Proceedings of ACM SIGCOMM '02*, available as *Computer Communications Review*, vol. 32:4, pp. 47-60, 2002. Journal version: *IEEE/ACM Transactions on Networking*, 12:5, pp. 767-780, 2004.

- “Optimal Plans for Aggregation,” with A. Broder. In *Proceedings of the 21st Annual ACM Symposium on Principles of Distributed Computing*, pp. 144-152, 2002.
- “Human-Guided Tabu Search,” with G. Klau, N. Lesh, and J. Marks. In *Proceedings of the 18th National Conference on Artificial Intelligence*, pp. 41-47, 2002.
- “The HuGS Platform: A Toolkit for Interactive Optimization,” with G. Klau, N. Lesh, J. Marks, and G. Schafer. In *Proceedings of the Workshop on Advanced Visual Interfaces*, pp. 324-330, 2002.
- “Exact Sampling of TCP Window States,” with A. Goel. In *Proceedings of IEEE INFOCOM*, pp. 259-265, 2002.
- “Balls and Bins Models with Feedback,” with E. Drinea and A. Frieze. In *Proceedings of the 11th Annual ACM-SIAM Symposium on Discrete Algorithms*, pp. 308-315, 2002.
- “A Brief History of Lognormal and Power Law Distributions.” In *Proceedings of the 39th Annual Allerton Conference on Communication, Control, and Computing*, pp. 182-191, 2001. Journal version: *Internet Mathematics*, 1(2):226-251, 2003.
- “Capacity Approaching Signal Constellations for Channels with Memory,” with A. Kavčić, X. Ma, and N. Varnica. In *Proceedings of the 39th Annual Allerton Conference on Communication, Control, and Computing*, pp. 311-320, 2001.
- “Compressed Bloom Filters.” In *Proceedings of the 20th Annual ACM Symposium on Principles of Distributed Computing*, pp. 144-150, 2001. Journal version: *IEEE/ACM Transactions on Networks*, 10:5, pp. 613-620, October 2002.
- “Towards More Completed Models of TCP Throughput,” with R. Rajaraman. *Journal of Supercomputing*, 20(2), pp. 137-160, September 2001.
- “Deriving Performance Bounds for ISI Channels Using Gallager Codes,” with A. Kavčić, B. Marcus, and B. Wilson. In *Proceedings of the 2001 International Symposium on Information Theory*, p. 345, 2001.
- “Toward Compressing Web Graphs,” with M. Adler. In *Proceedings of the 2001 Data Compression Conference*, pp. 203-212, 2001.
- “On the Hardness of Finding Multiple Preset Dictionaries.” In *Proceedings of the 2001 Data Compression Conference*, pp. 411-418, 2001. Journal version: *IEEE Transactions on Information Theory*, 50:7, pp. 1536-1539, 2004.
- “Fine-Grained Layered Multicast,” with J. Byers and M. Luby. In *Proceedings of IEEE INFOCOM 2001*, pp. 1143-1151, 2001. Journal version: “Fine-Grained Layered Multicast with STAIR,” with J. Byers, Gu-In Kwon, and M. Luby. To appear in *IEEE/ACM Transactions on Networking*.
- “Using Multiple Hash Functions to Improve IP Lookups,” with A. Broder. In *Proceedings of IEEE INFOCOM 2001*, pp. 1454-1463, 2001.
- “Improved Results for Route Planning in Stochastic Transportation Networks,” with J. Boyan. In *Proceedings of the 12th Annual ACM-SIAM Symposium on Discrete Algorithms*, pp. 895-902, 2001.

“Estimating Resemblance of MIDI Documents,” with S. Owen. In *Proceedings of the 3rd Workshop on Algorithm Engineering and Experiments*, available as *Lecture Notes in Computer Science 2153*, pp. 78-90, 2001.

“Linear Waste of Best Fit Bin Packing on Skewed Distributions,” with Claire Kenyon. In *Proceedings of the 41st Annual IEEE Symposium on Foundations of Computer Science*, pp. 582-589, 2000. Journal version: invited to special issue of *Random Structures and Algorithms*, 20(3), pp. 441-464, 2002.

“FLID/DL: Congestion Control for Layered Multicast,” with J. Byers, M. Frumin, G. Horn, M. Luby, A. Roetter, and W. Shaver. In *Proceedings of the 2nd International Workshop on Networked Group Communication (NGC 2000)*, pp. 71-81, 2000. Journal version: with J. Byers, G. Horn, M. Luby, and W. Shaver. *IEEE Journal on Selected Areas in Communications*, 20:8, pp. 1558-1570, October 2002.

“On Near-Uniform URL Sampling,” with M. Henzinger, A. Heydon, and M. Najork. In *Proceedings of the Ninth International World Wide Web Conference*, pp. 295-308, 2000.

“An Extension of Path Coupling and Its Application to the Glauber Dynamics for Path Coupling,” with M. Dyer, L. Goldberg, C. Greenhill, and M. Jerrum. In *Proceedings of the 11th Annual ACM-SIAM Symposium on Discrete Algorithms*, pp. 616-623, 2000. Journal version: *SIAM Journal on Computing*, 30(6), pp. 1962-1975, 2001.

“Improved Classification via Connectivity Information,” with A. Broder and R. Krauthgamer. In *Proceedings of the 11th Annual ACM-SIAM Symposium on Discrete Algorithms*, pp. 576-585, 2000.

“The Asymptotics of Selecting the Shortest of Two, Improved,” with B. Vöcking. In *Proceedings of the 37th Annual Allerton Conference on Communication, Control, and Computing*, pp. 326-327, 1999. Extended version appears as a chapter in *Analytic Methods in Applied Probability: In Memory of Fridrikh Karpelevich*, American Mathematical Society, pp. 165-176, 2002.

“Completeness and Robustness Properties of Min-Wise Independent Permutations,” with A. Broder. In *Proceedings of Random '99*, available as *Lecture Notes in Computer Science 1671*, pp. 1-10, 1999. Journal version: *Random Structures and Algorithms*, 18(1), pp. 18-30, January 2001.

“Measuring Index Quality Using Random Walks on the Web,” with M. Henzinger, A. Heydon, and M. Najork. In *Proceedings of the Eighth International World Wide Web Conference*, pp. 213-225, Elsevier Science, 1999.

“Analysis of Timing Based Mutual Exclusion with Random Times,” with E. Gafni. In *Proceedings of the 18th Annual ACM Symposium on Principles of Distributed Computing*, pp. 13-21, 1999. Journal version: *SIAM Journal on Computing*, 31:3, pp. 816-837, 2001.

“Accessing Multiple Mirror Sites in Parallel: Using Tornado Codes to Speed Up Downloads,” with J. Byers and M. Luby. In *Proceedings of the 18th Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM '99)*, pp. 275-284, 1999.

“Unscrambling Address Lines (Abstract),” with A. Broder and L. Moll. In *Proceedings of the 10th Annual ACM-SIAM Symposium on Discrete Algorithms*, pp. 870-871, 1999.

“A Derandomization Using Min-Wise Independent Permutations,” with A. Broder and M. Charikar. In *Proceedings of Random '98*, available as *Lecture Notes in Computer Science 1518*, pp. 15-24, October 1998. Journal version: *Journal of Discrete Algorithms*, 1:1, pp. 11-20, 2003.

“On Balls and Bins with Deletions,” with R. Cole, A. Frieze, B. M. Maggs, A. W. Richa, R. Sitaraman, and E. Upfal. In *Proceedings of Random '98*, available as *Lecture Notes in Computer Science 1518*, pp. 145-158, October 1998.

“Studying Balanced Allocations with Differential Equations.” *Combinatorics, Probability, and Computing*, vol. 8, pp. 473-482, 1999.

“Delayed Information and Action in On-Line Algorithms,” with S. Albers and M. Charikar. In *Proceedings of the 39th Annual IEEE Symposium on Foundations of Computer Science (FOCS-98)*, pp. 71-80, November 1998. Journal version: *Information and Computation*, 170, pp. 135-152, 2001.

“A Digital Fountain Approach to the Reliable Distribution of Bulk Data,” with John Byers, Michael Luby, and Ashu Rege. In *Proceedings of ACM SIGCOMM '98*, available as *Computer Communications Review*, vol. 28:4, pp. 56-67, 1998. Journal version: *IEEE Journal on Selected Areas in Communication*, 20:8, pp. 1528-1540, October 2002.

“Analyses of Load Stealing Models Using Differential Equations.” In *Proceedings of the 10th Annual ACM Symposium on Parallel Algorithms and Architectures*, pp. 212-221, 1998. Journal version: *Theory of Computing Systems*, 34:1, pp. 77-98, January 2001.

“Load Adaptive Routing on Butterfly Networks,” with R. Cole, B. Maggs, F. Meyer auf der Heide, A. Richa, K. Schröder, R. Sitaraman, and B. Vöcking. In *Proceedings of the 30th Annual ACM Symposium on Theory of Computing (STOC-98)*, pp. 378-388, ACM Press, May 23-26, 1998.

“Min-Wise Independent Permutations,” with Andrei Broder, Moses Charikar, and Alan Frieze. In *Proceedings of the 30th Annual ACM Symposium on Theory of Computing (STOC-98)*, pp. 327-336, ACM Press, May 23-26, 1998. Journal version: *Journal of Computer and System Sciences*, 60(3), pp. 630-659, 2000.

“Analysis of Low Density Codes and Improved Designs Using Irregular Graphs,” with Michael Luby, Amin Shokrollahi, and Dan Spielman. In *Proceedings of the 30th Annual ACM Symposium on Theory of Computing (STOC-98)*, pp. 249-258, ACM Press, May 23-26, 1998.

“Improved Low Density Parity Check Codes Using Irregular Graphs and Belief Propagation,” with Michael Luby, Amin Shokrollahi, and Dan Spielman. In *Proceedings of the 1998 International Symposium on Information Theory*. Extended version available as Compaq Systems Research Center Technical Note 1998-009. Journal version: “Improved Low Density Parity Check Codes Using Irregular Graphs,” *IEEE Transactions on Information Theory*, 47(2), pp. 585-598, 2001.

“Analysis of Random Processes via And-Or Tree Evaluation,” with Michael Luby and Amin Shokrollahi. In *Proceedings of the 9th Annual ACM-SIAM Symposium on Discrete Algorithms*, pp. 364-373, 1998.

“Average Case Analyses of First Fit and Random Fit Bin Packing,” with Susanne Albers. In *Proceedings of the 9th Annual ACM-SIAM Symposium on Discrete Algorithms*, pp.

290-299, 1998. Journal version: *Random Structures and Algorithms*, 16(3), pp. 240-259, 2000.

“How Useful is Old Information?” In *Proceedings of the 16th ACM Symposium on Principles of Distributed Computing*, pp. 83-91, 1997. Journal version: *IEEE Transactions on Parallel and Distributed Systems*, 11:1, pp. 6-20, January 2000.

“Revisiting the COUNTER Algorithms for List Update,” with Susanne Albers. *Inf. Proc. Letters*, 64, pp. 155-160, 1997.

“Practical Loss-Resilient Codes,” with Michael Luby, Amin Shokrollahi, Dan Spielman, and Volker Stemann. In *Proceedings of the 29th ACM Symposium on the Theory of Computing*, pp. 150-159, 1997. Journal version: “Efficient Erasure Correcting Codes,” with Michael Luby, Amin Shokrollahi, and Dan Spielman. *IEEE Transactions on Information Theory*, 47(2), pp 569-584, 2001.

“On the Analysis of Randomized Load Balancing Schemes.” In *Proceedings of the 9th ACM Symposium on Parallel Algorithms and Architectures*, pp. 292-301, 1997. Journal version: special conference issue of *Theory of Computing Systems*, vol. 32, pp. 361-386, 1999.

“Load Balancing and Density Dependent Jump Markov Processes.” In *Proceedings of the 37th Annual Symposium on Foundations of Computer Science*, pp. 213-222, 1996. Journal version: “The Power of Two Choices in Randomized Load Balancing.” *IEEE Transactions on Parallel and Distributed Computing*, 12(10), pp. 1094-1104, 2001.

“Designing Stimulating Programming Assignments for an Algorithms Course: A Collection of Problems Based on Random Graphs.” In *SIGCSE Bulletin*, 28:3, pp. 29-36, September 1996.

“Constant Time per Edge is Optimal in Rooted Tree Networks.” In *Proceedings of the 8th ACM Symposium on Parallel Algorithms and Architectures*, pp. 162-169, 1996. Journal version: *Distributed Computing* vol. 10:4, 1997, pp. 189-197.

“Average Case Analyses of List Update Algorithms, with Applications to Data Compression,” with Susanne Albers. In *Proceedings of the 1996 International Colloquium on Automata, Languages, and Programming*, pp. 514-525. Journal version: *Algorithmica*, vol. 21:3, 1998, pp. 312-329.

“Pattern-based Compression of Text Images,” with Andrei Broder. In *1996 Data Compression Conference*, pp. 300-309.

“Parallel Randomized Load Balancing,” with Micah Adler, Soumen Chakrabarti, and Lars Rasmussen. In *Proceedings of the 27th ACM Symposium on the Theory of Computing*, pp. 238-247, 1995. Journal version: *Random Structures and Algorithms*, vol. 13:2, September 1998, pp. 159-188.

“Bounds on the Greedy Routing Algorithm in Array Networks.” In *Proceedings of the 6th ACM Symposium on Parallel Algorithms and Architectures*, pp. 346-353, 1994. Journal version: special conference issue of the *Journal of Computer and System Sciences* vol. 53:3, December 1996, pp. 317-327.

“Computational Complexity of Loss Networks,” with Graham Louth and Frank Kelly. *Theoretical Computer Science*, vol. 125, May 1994, pp. 45-59.

Other Publications

“Lyra: An Efficient and Expressive Subquadratic Architecture for Modeling Biological Sequences,” with Krithik Ramesh, Sameed Siddiqui, Albert Gu, and Pardis Sabeti. Available at <https://arxiv.org/abs/2503.16351>.

“Juniper: Reconstructing Transmission Events from Next-Generation Sequencing Data at Scale,” with Ivan Specht, Gage Moreno, Taylor Brock-Fisher, Lydia Kraslinkova, Brittany Petros, Jonathan Pekar, Mark Schifferli, Ben Fry, Catherine Brown, Lawrence Madoff, Megan Burns, Stephen Schaffner, Daniel Park, Bronwn MacInnis, Al Ozonoff, Patrick Varilly, and Pardis Sabeti. Available at <https://www.medrxiv.org/content/10.1101/2025.03.02.25323192v1>.

“Delphy: scalable, near-real-time Bayesian phylogenetics for outbreaks,” with Patrik Varilly, Mark Schifferli, Katherine Yang, Tim Burcham, Paul Cronan, Olivia Glennon, Olivia Jacks, Ellory Laning, Libby Marrs, Kyle Oba, Shannon Yeung, Edyth Parker, Ifeanyi Omah, Jonathan E. Pekar, Laura Luebbert, Kristian G. Andersen, Daniel J. Park, Stephen F. Schaffner, Bronwyn L. MacInnis, Christian Happi, Jacob E. Lemieux, Al Ozonoff, Michael D. Mitzenmacher, Ben Fry, and Pardis C. Sabeti. Available at <https://www.biorxiv.org/content/10.1101/2025.03.25.645253v1>.

“Use of popularity information to reduce risk posed by guessing attacks,” with Stuart Schechter and Cormac Herley. US Patent 08959644. Issued 2/17/2015.

“Method and apparatus providing network coding based flow control,” with Jay Kumar Sundararajan, Devavrat Shah, Muriel Medard, Joao Barros, and Szymon Jakubczak. US Patent 08526451. Issued 9/3/2013.

“Method and apparatus providing network coding based flow control,” with Jay Kumar Sundararajan, Devavrat Shah, Muriel Medard, Joao Barros, and Szymon Jakubczak. US Patent 08130776. Issued 3/6/2012.

“Method for finding optimal paths using a stochastic network model,” with Evdokia Nikolova and Matthew Brand. US Patent 07573866. Issued 8/11/2009.

“Ranking search engine results,” with Monika Henzinger. US Patent 07541388. Issued 11/11/2008.

“Distributed, compressed Bloom filter Web cache server.” US Patent 06920477. Issued 7/19/2005.

“Method for packing rectangular strips,” with Neal Lesh and Joe Marks. US Patent 06832129. Issued 12/14/2004.

“System and method for near-uniform sampling of web page addresses,” with Monika Henzinger, Allan Heydon, and Marc Najork. US Patent 06594694. Issued 7/15/2003.

“On demand encoding with a window,” with Michael Luby, Gavin Horn, Jeffrey Persch, John Byers, and Armin Haken. US Patent 06486803. Issued 11/26/2002.

“Generating high weight encoding symbols using a basis,” with Armin Haken, Michael Luby, Gavin Horn, Diane Hernek, and John Byers. US Patent 06411223. Issued 6/25/2002.

“Space-efficient multi-cycle barrel shifter circuit,” with Laurent Moll. US Patent 06314156. Issued 11/6/2001.

“Method for determining a random permutation of variables by applying a test function,” with Andrei Broder, Laurent Moll, and Mark Shand. US Patent 06292762. Issued 9/18/2001.

“Loss resilient code with double heavy tailed series of redundant layers,” with Michael Luby. US Patent 06195777. Issued 2/27/2001.

“Message encoding with irregular graphing,” with Michael Luby, Amin Shokrollahi, Dan Spielman, and Volker Stemann. US Patent 06163870. Issued 12/19/2000.

“Compression of grey scale images of text,” with Andrei Broder. US Patent 06088309. Issued 7/11/2000.

“Irregularly graphed encoding technique,” with Michael Luby, Amin Shokrollahi, Dan Spielman, and Volker Stemann. US Patent 06081909. Issued 6/27/2000.

“Loss resilient decoding technique,” with Michael Luby, Amin Shokrollahi, Dan Spielman, and Volker Stemann. US Patent 06073250. Issued 6/6/2000.

“Task processing optimization in a multiprocessor system,” with Andrei Broder. US Patent 05991808. Issued 11/23/1999.

“Compression protocol with multiple preset dictionaries,” with Andrei Broder and Jeff Mogul. US Patent 05953503. Issued 9/14/1999.

“When Simple Hash Functions Suffice,” with K. Chung and S. Vadhan. Appears as a chapter in *Beyond Worst Case Analysis*, edited by Tim Roughgarden, Cambridge University Press.

“Algorithms with Predictions,” with S. Vassilvitskii. Appears as a chapter in *Beyond Worst Case Analysis*, edited by Tim Roughgarden, Cambridge University Press.

“Some Practical Randomized Algorithms and Data Structures,” with E. Upfal. Appears as a chapter in *Computing Handbook, Third Edition: Computer Science and Software Engineering*, edited by T. Gonzalez, J. Diaz-Herrera, and A. Tucker. Chapman and Hall, 2014.

“Building an Efficient Hash Table on the GPU,” with D. Alcantara, V. Volkov, S. Sengupta, J. Owens, and N. Amenta. Appears as a chapter in *GPU Computing Gems: Jade Edition*, edited by Wen-mei W. Hwu. Morgan Kaufmann, 2011.

“Codes – Protecting Data Against Loss and Errors.” Appears as a chapter in *Algorithms Unplugged*, edited by B. Vöcking, M. Dietzfelbinger, R. Reischuk, and H. Vollmer. Springer, 2010.

“Hash-Based Techniques for High-Speed Packet Processing”, with A. Kirsch and G. Varghese. Appears as a chapter in *Algorithms for Next Generation Networks*, edited by G. Cormode and M. Thottan. Springer, 2010.

“Bloom Filters.” *Encyclopedia of Database Systems*, 2009.

“Unbiasing Random Bits.” *Dr. Dobbs Journal*, No. 335, pp. 101-104, April 2002.

“Good Hash Tables & Multiple Hash Functions.” *Dr. Dobbs Journal*, No. 336, May 2002.

“Challenging Students with Creative Assignments.” *SIGACT News*, 32(1):70-73, 2001.

“An Experimental Assignment on Random Processes.” *SIGACT News*, 32(1):74-78, 2001.

“The Power of Two Random Choices: A Survey of Techniques and Results,” with A. Richa and R. Sitaraman. Appears as a chapter in *Handbook of Randomized Computing: Volume 1*, edited by P. Pardalos, S. Rajasekaran, and J. Rolim, pp. 255-312.

“Tornado Codes for Forward Error Correction,” with Michael Luby, in *Digital Forefront*.

“Bounds and Improvements for BiBa Signature Schemes,” with A. Perrig. Harvard Computer Science Group Technical Report TR-02-02.

“A Note on Low Density Parity Check Codes for Erasures and Errors.” Digital SRC Technical Note 1998-017.

“Tight Thresholds for the Pure Literal Rule.” Digital SRC Technical Note 1997-011.

“The Power of Two Choices in Randomized Load Balancing.” Ph.D. thesis, U.C. Berkeley, August 1996.

“*Calculus: A Solution Manual*,” with others, Harcourt Brace Jovanovich, 1993.

“*Elliptic Curves in Computer Science: Primality Testing, Factoring, and Cryptography*,” senior honors thesis, Harvard University, 1991. Published as a Hoopes Prize winner.

**Conference
Presentations**

“Queueing, Predictions, and LLMs: Challenges and Open Problems.” Tutorial, SIGMETRICS. June 2025.

“SkipPredict: When to Invest in Predictions for Scheduling. Presented at the TTIC Workshop on Learning-Augmented Algorithms, August 2024.

“Algorithms with Predictions.” Invited talk presented at the INFORMS 2023 Annual Meeting, October 2023.

“Algorithms with Predictions.” Presented at the 2nd Vienna Workshop on Computational Optimization. March 2023.

“Algorithms with Predictions.” Presented at the Sublinear Algorithms Workshop, FODSI/MIT. August 2022.

“Algorithms with Predictions: State of the Area.” Presented at the Algorithms with Predictions Workshop at the 54th Annual ACM Symposium on Theory of Computing. June 2022. Also panel lead.

“Efficient Randomized Network Telemetry Algorithms.” Presented at the Data plane algorithms in programmable networks tutorial at ACM SIGMETRICS 2022.

“Scheduling with Predictions and the Price of Misprediction.” Presented at the 11th Annual Innovations in Theoretical Computer Science Conference. January 2020.

“Learning-Augmented Algorithms: How ML Can Lead to Provably Better Algorithms.” ALGO Keynote Talk, September 2019.

“Robust Set Reconciliation via Locality Sensitive Hashing.” Presented at the 38th Symposium on Principles of Database Systems, July 2019.

“Arithmetic Progression Hypergraphs: Examining the Second Moment Method.” Presented at the 2019 ANALCO Workshop, January 2019.

“A Model for Learned Bloom Filters and Optimizing by Sandwiching.” Presented at the NeurIPS 2018 Poster Session.

“Adaptive Cuckoo Filters.” Presented at the 2018 ALENEX Conference, January 2018.

“Analyzing Distributed Join-Idle-Queue: A Fluid Limit Approach.” Presented at the 54th Annual Allerton Conference on Communication, Control, and Computing, October 2016.

“Voronoi Choice Games.” Presented at the 2016 International Colloquium on Automata, Languages, and Programming, July 2016.

“A New Approach to Analyzing Robin Hood Hashing.” Presented at the 2016 ANALCO Workshop, January 2016.

“More Analysis of Double Hashing for Balanced Allocations.” Presented at the 2016 ANALCO Workshop, January 2016.

“Data exploration: equitability and the maximal information coefficient.” Presented at the Workshop for Data-driven Algorithmics, September 2015.

“Multi-Party Set Reconciliation Using Characteristic Polynomials.” Presented at the 2014 Allerton Conference, October 2014.

“Parallel Peeling Algorithms.” Presented at the 2014 Symposium on Parallelism in Algorithms and Architectures, June 2014.

“Balanced Allocations and Double Hashing.” Presented at the 2014 Symposium on Parallelism in Algorithms and Architectures, June 2014.

“Peeling Arguments and Double Hashing.” Presented at the 2012 Allerton Conference, October 2012.

“The Complexity of Object Reconciliation and Open Problems Related to Set Difference and Coding.” Presented at the 2012 Allerton conference, October 2012.

“The Ecology of Communication, Control, and Computing Research and Education (Panel).” Presented at the 2012 Allerton conference, October 2012.

“Peeling Arguments: Invertible Bloom Lookup Tables and Biff Codes.” Presented at the 2012 Workshop on Algorithms for Modern Massive Data Sets (MMDS), July 2012.

“Invertible Bloom Lookup Tables.” Presented at the 2011 Allerton conference, September 2011.

“Brief Announcement: Large-Scale Multimaps.” Presented at the 2011 Symposium on Parallel Algorithms and Architectures, June 2011.

“On the Zero-Error Capacity Threshold for Deletion Channels.” Presented at the 2011 ITA Workshop, February 2011.

“Some Recent Uses of Hashing for Network Applications.” Presented at the 2010 Workshop on Research and Analysis of Tail Phenomena, August 2010.

“Some Recent Uses of Hashing for Network Applications.” Presented at the 2010 Workshop on Random Graphs and Algorithms, June 2010.

“Tight Asymptotic Bounds for the Deletion Channel with Small Deletion Probabilities.” Presented at the 2010 International Symposium on Information Theory, June 2010.

“Some Open Questions on Cuckoo Hashing.” Presented at the 2009 European Symposium on Algorithms, August 2009.

“On the Performance of Multiple Choice Hash Tables with Moves on Deletions and Inserts.” Presented at the 2008 Allerton conference, September 2008.

“Designing Floating Codes for Expected Performance.” Presented at the 2008 Allerton conference, September 2008.

“A Survey of Results for Deletion Channels and Related Synchronization Channels.” Presented at the 11th Scandinavian Workshop on Algorithm Theory, July 2008.

“Why Simple Hash Functions Work: Exploiting the Entropy in a Data Stream.” Presented at the 19th Annual ACM-SIAM Symposium on Discrete Algorithms, January 2008.

“Using a Queue to De-amortize Cuckoo Hashing in Hardware.” Presented at the 2007 Allerton conference, September 2007.

“Capacity Upper Bounds for Deletion Channels.” Presented at the 2007 Int’l Symposium on Information Theory (ISIT), June 2007.

“Codes for Deletion and Insertion Channels with Segmented Errors.” Presented at the 2007 Int’l Symposium on Information Theory (ISIT), June 2007.

“Digital Fountains: Applications and Related Issues.” Presented at the INFORMS Annual Meeting, November 2006.

“New Directions for Power Law Research.” Presented at the Radcliffe Exploratory Seminar on Dynamic Networks, October 2006.

“Bloom Filters via d -left Hashing and Dynamic Bit Reassignment.” Presented at the 2006 Allerton conference, September 2006.

“Beyond Bloom Filters: From Approximate Membership Checks to Approximate State Machines.” Presented at the 2006 SIGCOMM conference, September 2006.

“An Improved Construction for Counting Bloom Filters.” Presented at the European Symposium on Algorithms, September 2006.

“Building a Better Bloom Filter.” Presented at the European Symposium on Algorithms, September 2006.

“On the Theory and Practice of Data Recovery with Multiple Versions.” Presented at the 2006 International Symposium on Information Theory, July 2006.

“New Results for an Old Data Structure: Bloom Filters and Beyond.” Presented at the 2006 Stochastic Networks conference, June 2006.

“New Results and Open Problems for Deletion Channels.” Presented at ANALCO 2006, January 2006 (plenary talk).

“Digital Fountains and their Application to Informed Content Delivery over Adaptive Overlay Networks.” Presented at 19th International Symposium on Distributed Computing, September 2005 (plenary talk).

“Simple Summaries for Multiple Choice Hashing.” Presented at the 2005 Allerton Conference.

“New Directions in Power Law Research.” MSRI Workshop on Models of Real-World Random Networks, April 2005.

“Multi-dimensional Balanced Allocations.” Presented at the 16th Annual ACM-SIAM Symposium on Discrete Algorithms, January 2005.

“Digital Fountains: A Survey and Look Forward.” Presented at the 2004 Information Theory Workshop. October 2004.

“Improved Lower Bounds on the Capacity of I.I.D. Deletion Channels.” Presented at the 2004 Allerton Conference.

“Geometric Generalizations of the Power of Two Choices.” Presented at the 16th ACM Symposium on Parallel Algorithms and Architectures, June 2004.

“New Models and Methods for File Size Distributions.” Presented at the 40th Annual Allerton Conference on Communication, Control, and Computing, October 2003.

“Verification Codes for Deletion Channels.” Presented at the International Symposium on Information Theory, July 2003.

“New Exhaustive, Heuristic, and Interactive Algorithms for 2-Dimensional Bin-Packing.” Presented at the 2nd Annual Workshop on Random Graphs and Algorithms, June 2003.

“Network Applications of Bloom Filters: A Survey.” Presented at the 40th Annual Allerton Conference on Communication, Control, and Computing, October 2002.

“Verification Codes.” Presented at the 40th Annual Allerton Conference on Communication, Control, and Computing, October 2002.

“New Directions in Balls and Bins Problems.” Invited talk for the 6th International Workshop on Randomization and Approximation Methods in Computer Science. September 2002.

“Optimal Plans for Aggregation.” Presented at the 21st Annual ACM Symposium on Principles of Distributed Computing, July 2002.

“How to Get Across Town by Bus.” Presented at the INFORMS Annual Meeting, November 2001.

“A Brief History of Lognormal and Power Law Distributions.” Presented at the 39th Annual Allerton Conference on Communication, Control, and Computing, October 2001.

“Compressed Bloom Filters.” Presented at the 20th Annual ACM Symposium on Principles of Distributed Computing, August 2001.

“How Useful Is Old Information?” Presented at the INFORMS Applied Probability Conference, July 2001.

“Using Multiple Hash Functions to Improve IP Lookups.” Presented at the INFOCOM, April 2001.

“On the Hardness of Finding Multiple Preset Dictionaries.” Presented at the 2001 Data Compression Conference, March 2001.

“Improved Results for Route Planning in Stochastic Transportation Networks.” Presented at the 12th Annual ACM-SIAM Symposium on Discrete Algorithms, January 2001.

“Estimating Resemblance of MIDI Documents.” Presented at the 3rd Workshop on Algorithm Engineering and Experiments, January 2001.

“Linear Waste of Best Fit Bin Packing on Skewed Distributions.” Presented at the 41st Annual IEEE Symposium on Foundations of Computer Science, November 2000.

“Improved Classification via Connectivity Information.” Presented at the 11th Annual ACM Symposium on Discrete Algorithms, January 2000.

“The Asymptotics of Selecting the Shortest of Two, Improved.” Presented at the 37th Annual Allerton Conference on Communication, Control, and Computing, August 1999.

“Analysis of Timing Based Mutual Exclusion with Random Times.” Presented at the 18th Annual ACM Symposium on Principles of Distributed Computing, May 1999.

“Accessing Multiple Mirror Sites in Parallel: Using Tornado Codes to Speed Up Downloads.” Presented at INFOCOM ’99, March 1999.

“Delayed Information and Action in On-Line Algorithms.” Presented at the 39th Annual IEEE Symposium on Foundations of Computer Science, November 1998.

“A Derandomization Using Min-Wise Independent Permutations.” Presented at Random ’98, October 1998.

“On Balls and Bins with Deletions.” Presented at Random ’98, October 1998.

“The Power of Two Choices and Other Examples of Using Differential Equations to Analyze Algorithms.” Invited presentation at the Warwick Randomized Algorithms and Stochastic Simulation Tutorial and Workshop. July 1988.

“Analysis of Low Density Codes and Improved Designs Using Irregular Graphs.” Presented at the 30th Annual ACM Symposium on Theory of Computing, May 1998.

“Average Case Analyses of First Fit and Random Fit Bin Packing.” Presented at the 9th Annual ACM Symposium on Discrete Algorithms, January 1998.

“How Useful Is Old Information?” Presented at the 16th Annual ACM Symposium on Principle of Distributed Computing, August 1997.

“On the Analysis of Randomized Load Balancing Schemes.” Presented at the 9th ACM Symposium on Parallel Algorithms and Architectures, Newport, Rhode Island, June 1997.

“Load Balancing and Density Dependent Jump Markov Processes.” Presented at the 37th Annual Symposium on Foundations of Computer Science, Burlington, Vermont, October 1996.

“Constant Time per Edge is Optimal in Rooted Tree Networks.” Presented at the 8th ACM Symposium on Parallel Algorithms and Architectures, Padua, Italy, June 1996.

“GRID: Pattern-based Compression of Text Images.” Presented at the 1996 Data Compression Conference, Snowbird, Utah, April 1996.

“Parallel Randomized Load Balancing.” Presented at the 27th ACM Symposium on the Theory of Computing, Las Vegas, Nevada, May 1995. Extended version presented at the Bay Area Theory Seminar, July 1995.

“Bounds on the Greedy Routing Algorithm on Array Networks.” Presented at the 6th ACM Symposium on Parallel Algorithms and Architectures, Cape May, New Jersey, June 1994.

Other Presentations

“Algorithms with Predictions in Scheduling.” Johns Hopkins University. 2025.

“Algorithms with Predictions.” Georgia Tech. 2023.

“Modern Algorithms Lecture Series.” Lipari Summer School. 2022.

“Algorithms with Predictions.” Carnegie Mellon, UC Santa Barbara, Technical University of Munich. 2022.

“Locality Sensitive Hashing.” Broad Institute (Models, Inference, and Algorithms).

“Algorithms with Predictions.” Open Data Science Conference East 2020, Open Data Science Conference Europe 2020, Foundations of Data Science Institute Kickoff Workshop 2020.

“Learning-Augmented Algorithms: How ML Can Lead to Provably Better Algorithms.” Rice University. 2019.

“Metric Sublinear Algorithms via Linear Sampling.” Sublinear Algorithms Workshop (MIT), Baidu Research. 2018.

“Bloom Filters, Cuckoo Hashing, Cuckoo Filters, and Adaptive Cuckoo Filters.” Northeastern University, Univ. of Illinois at Urbana-Champaign, Georgetown University, University of California Irvine, Baidu Research, Georgia Tech. 2018.

“Hashing Lecture Series.” Swedish Summer School in Computer Science. 2016.

“More Analysis of Double Hashing for Balanced Allocations.” U.C. Irvine. 2016.

“Algorithm Engineering with Hashing,” U.C. Irvine, AMD. 2016.

“Bloom Filters and Such,” EADS Summer School on Hashing: Theory and Applications, July 2014.

“Cuckoo Hashing, Balanced Allocations, and Such,” EADS Summer School on Hashing: Theory and Applications, July 2014.

“Peeling Arguments: Invertible Bloom Lookup Tables and Biff Codes,” ICERM Workshop on Stochastic Graph Models, March 2014.

“Peeling Arguments: Invertible Bloom Lookup Tables and Biff Codes,” University of Copenhagen. 2013.

“Some Hash-Based Data Structures and. Algorithms Everyone Should Know.” UC Davis, 2013.

“Peeling Algorithms,” Simons Institute. 2013.

“Verifying Computations in the Cloud (and Elsewhere),” TTI/Vanguard : Ginormous Systems. Microsoft Faculty Summit. 2013.

“The Groupon Effect on Yelp Ratings: a Root Cause Analysis,” Cornell, 2012.

“Peeling Arguments: Invertible Bloom Lookup Tables and Biff Codes,” Google. 2012.

“Detecting Novel Associations in Large Data Sets.” eHarmony (LA Machine Learning). 2012.

“Cuckoo Hashing: New Results and Open Questions.” University of Madison-Wisconsin. 2011.

“Practical Verified Computation with Streaming Interactive Proofs.” University of Madison-Wisconsin. 2011.

“The Power of Choice: Simple Hash-Based Data Structures Everyone Should Know.” Santa Fe Institute. 2011.

“Information Asymmetries in Pay-Per-Bid Auctions: How Swoopo Makes Bank.” Liverpool University, Microsoft UK. 2010.

“Some Open Questions on Cuckoo Hashing.” Stanford, Microsoft Research, Yale, Dartmouth, Princeton, Cambridge University, University College London . 2009-2010.

“An Efficient, Rigorous Approach for Identifying Statistically Significant Frequent Itemsets.” Google. 2009.

“Some Recent Results on Cuckoo Hashing.” AT&T. 2009.

“Some Results on Coding for Flash Memory.” University College London. 2009.

“Network Coding Meets TCP.” University College London. 2009.

“A Survey of Results for Deletion Channels and Related Synchronization Channels.” Cambridge, Georgia Tech, Microsoft Research, UCLA, 2009-2010.

“Bloom Filters, Related Data Structures, and their Applications.” Cambridge (Kuwait Lecture). 2009.

“A History of and New Directions for Power Law Research.” SUNY Buffalo. 2009.

“Why Simple Hash Functions Work.” Cisco. 2008.

“Cuckoo Hashing and CAMs.” Cisco, Google. 2008.

“New Results in Hash-Based Data Structures.” Cisco. 2007.

“The Hiring Problem and Lake Wobegon Strategies.” Microsoft, U.C. Berkeley, Princeton, Google, Yahoo. 2007-2008.

“A Brief History of Lognormal and Power Law Distributions.” Harvard Statistics. 2006.

“New Results and Open Problems for Deletion Channels and Related Synchronization Problems.” CMU, Brown, Harvard. 2006.

“Some New Results on an Old Data Structure, Bloom Filters .” Stanford, Cisco, Microsoft. 2005.

“Improved Lower Bounds for Deletion Channels.” MIT, EPFL (Lausanne). 2005.

“Building a Better Bloom Filter.” Google, Tufts, EPFL (Lausanne), MIT. 2004-05.

“Geometric Generalizations of the Power of Two Choices.” Columbia, Brown, MIT, Georgia Tech. 2004-05.

“New Lower Bounds on the Capacity of the Binary Deletion Channel.” MIT, 2004.

“Digital Fountains, and their Use for Informed Content Delivery across Overlay Networks.” Cornell, 2003.

“Verification Codes for Error and Deletion Channels.” U.C. Berkeley, 2003.

“New Exhaustive, Heuristic, and Interactive Algorithms for 2D Strip Packing.” U.C. Berkeley, 2003.

“Dynamic Models for File Sizes and Double Pareto Distributions.” IBM Research, Microsoft Silicon Valley, Stanford, Tufts, Boston University, Random Graphs and Algorithms Workshop. 2002.

“Verification Codes.” DIMACS workshop. 2001.

“Getting Across Town by Bus, Aggregation, and Reliability.” MIT (LIDS), Mitsubishi Electronic Research Labs. 2001.

“New Ideas for Layered Multicast Congestion Control.” International Computer Science Institute. 2001.

“Compressed Bloom Filters, and Towards Compressing Web Graphs.” U. of Washington, AT&T Labs, U. C. Berkeley, MIT, Brandeis, Carnegie Mellon University, Stanford, DIMACS workshop on compression, U.C. San Diego. 2001.

“Toward Compressing Web Graphs.” Compaq Systems Research Center, Altavista Corporation. 2000.

“Improved Results for Route Planning in Stochastic Transportation Networks.” Compaq Systems Research Center. 2000.

“Choosing the Shortest of Two, Improved, with Applications to IP Routing.” U. Washington, Princeton, Yale, U.C. Berkeley, Boston University, Stanford, M.I.T., Northeastern, Compaq Systems Research Center. 2000.

“Tornado Codes, with Applications to Reliable Multicast.” Harvard, CMU, Duke, AT&T Labs, Univ. of Illinois at Urbana-Champaign, University of Rhode Island, Boston University. 1998-2000.

“How Useful Is Old Information?” MIT, U.C. Berkeley, Stanford, Microsoft. 1997.

“Practical Loss-Resilient Codes.” U.C. Berkeley, Harvard. 1997.

“An Average Case Analysis for First-Fit Bin Packing.” Stanford, DIMACS workshop. 1997.

“The Power of Two Choices.” U.C. Berkeley, Brown, Harvard, Boston University, IBM Almaden, Digital Systems Research Center. 1997.

“The Power of Two Choices and Other Examples of Using Differential Equations to Analyze Algorithms.” U. Mass.