From Digital Exceptionalism to Computational Scarcity

Jean-François Blanchette

A common explanation for the extraordinary success of the Internet is digital exceptionalism. In essence, it contends thtat we have been blessed with the realization of the dream of which our ancestors caught only imperfect glimpses. Parchment, printing, telegraph, radio, television: each of these technologies led in their own times to their own information revolutions. This, however, is it, the genuine artifact, The Information Age—capital T, capital I, capital A.[01]

This particular sublimity of our times hinges on two distinct assumptions: first, in the words of Nicholas Negroponte, we have finally moved "from atoms to bits."[02] Reduced to its purest representation as 0s and 1s, information has finally reached the end of its historical process of emancipation from the material—*le degré zéro de l'information*. Bits are wholly indifferent to their particular media carrier (whether hard drive, optical disk, or network wires) and to the particular signal that encodes them (whether magnetic polarities, voltages, or pulses of light). As such, bits transcend the economics and logistics of analog media and are immune to the corruption, degradation, and decay that necessarily result from the handling of material carriers of information.

Second, in the words of Lawrence Lessig, Jonathan Zittrain, and Tim Wu, the Internet is uniquely "generative."[03] Through the modular principles that have guided its design, the Internet's highly interoperable architecture enables the next evolutionary step to the vertically integrated firm: a new and highly flexible market structure that provides a democratic "plug-and-play" environment within which innovators may unleash their creativity. The unique ability of modular architectures to render irrelevant the physical basis of computing allows innovation to bypass the drudgery of physical implementation and proceed on the basis of the creative assembly of high-level components.

According to digital exceptionalism, then, immateriality (or at least, freedom from the constraints of the material world) is fundamental to the ability of the digital to upend the analog world. This is why any media that can be digitized or produced digitally will eventually succumb to the logics of digital information and its circulation through electronic networks.

This putative superiority of our digital age is typically credited as the gift of two exceptional individuals, Alan Turing and Claude Shannon, who are celebrated as the founding fathers of the Internet in light of their contributions to the dematerialization of computing and telecommunications, respectively.[04] Both mathematicians proposed models—the Turing machine and the Shannon-Weaver model—that assimilate computation and communication to symbol manipulation; that is, both models formally manipulated a set of symbolic representations that have themselves been reduced to their most fundamental expression as 0s and 1s.[05] In the service of their theoretical arguments, both models eschew considerations of computational resources in the realization of these manipulations. The Turing machine, for example, enjoys the luxury of infinite storage space, and its operation consumes no power whatsoever.[06]

The lesson of the Turing machine is that software is ontologically superior to hardware, while the lesson of the Shannon-Weaver model is that, no matter how much noise affects the communication channel, bits can always be restored to their original purity. In both cases, the networked computer is a technology that has become indifferent to its material instantiation and whose behavior is fundamentally mathematical. This behavior can be and has been implemented using a broad range of material technologies beyond the silicon circuits that dominate today. Indeed, working computers have been built out of tinker toys, and carrier pigeons have been proposed as a possible medium for the realization of the TCP/IP networking protocols.[07]

There are many reasons to be concerned with digital exceptionalism; among other things, it provides a poor foundation for computing pedagogy.[08] However, the dimension I will focus on here is that digital exceptionalism actively obscures an important technical and social reality, that networked computing is about the sharing of scarce resources. Contrast the idealization of the computer embodied by the Turing machine with its infinite storage and power supply and with various visualizations of computational resources (e.g., processing, energy, memory) offered by Apple's Activity Monitor application, resources that are clearly limited in nature and actively contested between applications.

The focus on the logical dimension of bits—their identity as 0s and 1s—negates their material dimension, their identity as physical entities, whether through light, electricity, or radio waves, or magnetized particles. Yet the debates that today dominate the political economy of computing—net neutrality, access to broadband, technological literacy, the politics of platforms and standardization—cannot be understood without accounting for the intricate relation between the logical and material dimension of bits and, in particular, how the computational resources that ensure the material realization of bits are, of course, always limited.

The key to the relationship between the logical and material dimension of bits lies in the much-too-obvious fact that the computer is itself computerized. That is, the symbolic manipulations performed by computers concern not only representations of the external world but also representations of their own material resources (e.g., a file, or a TCP/IP packet) as well. These representations are located not at the level of software applications, per se, but at the level of the computing infrastructure. It is at that level that computing's complicated relationship to materiality is played out.

The Computing Infrastructure

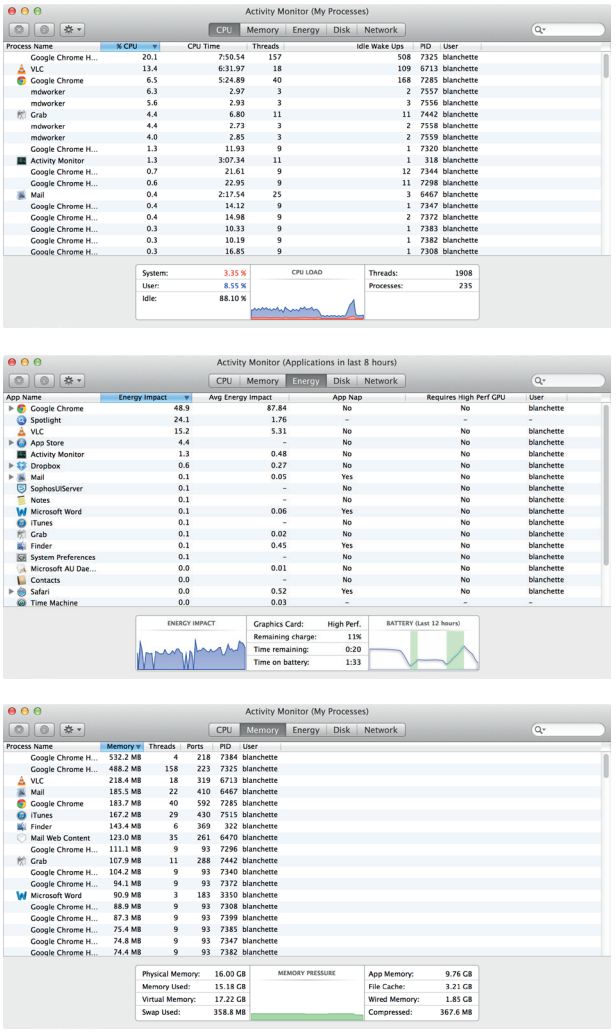
The evolution of computing technology is typically portrayed in terms of its extraordinary rates of growth in performance. Year after year, processors perform more instructions at ever-increasing speeds, storage devices are able to pack more bits into ever-smaller amounts of space, and network wires transmit more data at ever-faster rates. Indeed, much of our understanding of the extraordinary spread of computing in the past sixty years is based on the idea that the fundamental computing resources of processing power, storage, and bandwidth have and will continue to become simultaneously more powerful and cheaper.[09]

This particular frame of analysis, however, captures only a limited subset of the forces that have driven the evolution of networked computing. Less visible but equally essential dimensions include, on the one hand, modularity, the design technique used to break down complexity and manage the high rate of technological change characteristic of computing technologies, and on the other hand, the sharing and distribution of limited computing resources. Much of these dynamics take place somewhat out of sight, at the level of computing infrastructure, rather than at the level of applications, the more plainly visible space where users extract personal value from computing technologies. Only by examining these three shaping forces—performance increases, modular design, and sharing—together can the evolutionary dynamics of the computing ecosystem, including its current manifestation as the dematerialized Cloud, be analyzed.[10]

In computing, infrastructure can be defined quite simply as the elements of the computing ecosystem that provide services to applications (for example, that perform arithmetic functions, store and retrieve bits, send packets

over networks), in contrast to the applications that provide services to users (for example, that process words, or post a status update). The computing infrastructure is composed of both software and hardware: system abstractions such as file systems or packets, storage media such as flash or hard drives, communication protocols such as TCP/IP or ADSL. It provides the various components from which designers can build computing systems as diverse as the Google search engine, Microsoft Word running on a desktop, and the TiVo software. It is what allows computers to be multipurpose, while simultaneously managing the high rate of technical change of computing resources. These two characteristics of computing systems are easily taken for granted today, but they required considerable design innovation to come into being.

The first electromechanical computing machines could only perform a limited range of computational tasks. Herman Hollerith's first tabulator, developed for the 1890 census, was specifically tailored to add up census schedules. Early digital computers such as the ENIAC could be reconfigured to perform different types of computation only through a time-consuming rewiring of the various hardware components. The concept of the stored program, as formulated by John von Neumann in 1943, provided one elegant solution to the issue: the execution of the different components of computers could be directed by way of a sequence of instructions—a program. A single computer could perform distinct computational tasks merely by executing a different program, and while designing such programs proved to be no simple task, once written, programs could be switched in a matter of seconds.



Computational resources (processing, energy, memory) represented by Apple's Activity Monitor application.

Modularity

This newfound versatility came with some important trade-offs, however. Even as von Neumann and his coinventors gave birth to the software/hardware division, the two remained fused. In the early days of computing, hardly any distance separated program and hardware. The manual of operations for the first commercially produced stored program computer, the IBM 701, included such specific timing considerations as, "to keep the card reader in continuous motion, it is necessary to give the read instruction between 20 and 70 milliseconds after the 12-right copy instruction."[11] Phrased differently, programmers had to take into account specific characteristics of the hardware—in this case, the number of operations the processor would execute before the next card would be available for reading data or instructions from the punched card reader (the dominant input/output technology in the early 1950s). This rapidly became

problematic insofar as each succeeding generation of computers offered new, faster hardware, and programs had to be rewritten from scratch to take advantage of their new characteristics, at great expense of time and money.

By the 1960s the situation had become a sore point for the industry as a whole, prompting market leader IBM to seek a remedy for the issue. The solution consisted in designing a series of processor lines, each compatible with one another (including peripherals): any program written for one line would be able to execute on any other line, albeit with different levels of performance. To achieve such compatibility, IBM relied on modularity as a design strategy, where each component of the system was conceived as a discrete black box with a standardized interface.[12] In the resulting System/360, for example, all processors responded to the same set of instructions, even though their internal architecture might differ widely. For the first time, programmers could design programs with the confidence that, as long as components retained the same interface, programs would continue to run in spite of technical advances.

Compatibility proved much more than just an engineering feature, however, as it profoundly altered the economics of the computing ecosystem: components with standardized interfaces could just as well be produced by competitors, and many IBM engineers left the company to launch their own lines of cheaper compatible processors and peripherals. Modular computing systems rapidly ushered in an era of vertical disintegration of the industry and a new form of market organization. Indeed, at the core of Lessig, Zittrain, and Wu's "architecture is politics" argument is the acknowledgment that the extraordinary success of the Internet is a direct outcome of its modular architecture, which effectively lowers barriers to entry for prospective market participants and fosters experimentation at the component level.

Sharing and Virtualization

A parallel historical development involved the emergence of operating systems and virtualization. In the early days of computing, computing speed was hampered by two main bottlenecks. On the one hand, programs were loaded and executed sequentially, in a slow and cumbersome process termed "batch processing": at any given time, only one user could access the machine's expensive resources (processor, storage, etc.). On the other hand, increases in processing speed were limited by the extremely slow speed of storage technologies: program execution was often stalled as the processor waited for data to be loaded or written to storage.

Above: Server maintenance at Facebook's Prineville Data Center.

A decisive breakthrough came with the invention of time-sharing. Instead of executing sequentially, multiple programs were loaded simultaneously and executed under the authority a new program—the supervisor. In similar fashion to real estate time-sharing, the supervisor amortizes an important capital expense (the processor) by distributing it among multiple noncompeting users. By allocating to each program a slice of processing time and by circling rapidly in round-robin fashion between them, each user enjoyed the illusion of having full control of the computer's resources, although in effect they had that control for only a small fraction of the time. Because the supervisor could use the time previously spent waiting for storage devices to service other users, individual performance did not suffer overall. In today's terms, time-sharing virtualized the processor: it created an abstraction of a computing resource (a whole processor, when only a portion was available) so that it could be more efficiently shared among multiple users. User programs directly interacted no longer with the processor but rather with the abstraction provided by the supervisor, which sliced and diced the actual processor among as many users as could be supported.

Just like modularity, time-sharing profoundly changed the economics of computing. By allowing more users to extract more usage out of the most expensive component of a computer (its processor), institutions were able to make the most out of their (increasingly large) capital investments. Through the design of appropriate abstractions, enormous gains in computing efficiency could be had by the efficient sharing of costly and limited computing resources. At the same time, the supervisor ushered in the era of operating systems—software that would serve as a mediating layer, to control applications' access to computing resources, whether processor, storage, or network.

Distributing Computing

Another strand of the history of computing design involves the distribution of computing resources in space. It addresses the question of where computing resources (processing, storage, data) should be located, in relationship to each other and to users. The criteria for choice includes not only computational efficiency but also crucial issues of control, cost, maintenance, reliability, security, and access, among others. During the relatively short history of the field, different architectures have successively dominated the landscape.

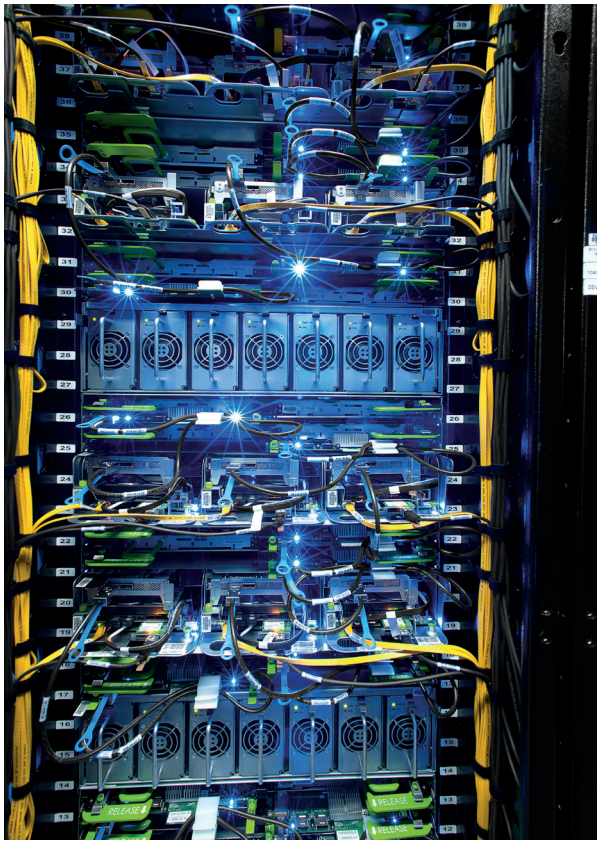
Early digital computers took the shape of mainframes: single-user machines in which all computing resources were centrally located, controlled, and maintained, often in the same room. Users accessed the mainframe through computer operators, who controlled available software and data, both of which had to be loaded on input/output devices located in the same physical space.

The advent of time-sharing dramatically transformed this setup: the virtualized processor of the mainframe was partitioned among multiple users, who accessed it through terminals (connected through either local wires or phone lines). The mainframe functioned as a server to these multiple clients, providing access to software and data stored either locally or remotely. At this stage the data transiting over the wires was textual: commands typed by users, and the results of their queries, typically textual and numeric information contained in databanks or the output of programs. Although access had been expanded, control remained local to the machine. Security became a new problem, as multiple users shared access to processors, storage, data, and programs.

Personal computing provided users with unprecedented control over their own processor, storage devices, software, and data. At the same time, personal computing introduced a host of challenges to the workplace: in contrast to the centralized mainframe, every employee's computer needed to be set up, maintained, upgraded, repaired, and provided with individual copies of software. In addition, processing and storage capacity were potentially wasted as individual machines sat idle by night. Personal computers also integrated with mainframes, insofar as they could be used as terminals to connect to institutional mainframes to access software or commercial services (databanks).

By allowing personal computers to connect with one another, the Internet has yet again broadened the scope of architectural possibilities for the distribution of computing resources. By and large, the dominant relationship has been one of client/server: users' devices download content (query results, for example, or streaming data) from the Cloud to local clients (primarily web browsers). Netflix serves as the paradigmatic example. This is, however, only one of many possible configurations for distributed computing. Peer-to-peer computing, in its many flavors, provides a vibrant example of an entirely different model for pooling distributed resources—one whose applications go far beyond mere illegal file sharing. Grid computing, as exemplified by the SETI@home project, leverages the idle cycles of thousands of machines to solve computationally intensive problems such a protein folding or climate modeling.

Today's age of mobile computing has emerged in symbiotic relationship with the Cloud. Given their limited storage, processing, and energy resources, portable



Server maintenance at Facebook's Prineville Data Center.

devices such as smartphones, tablets, and netbooks rely on cloud services to provide the required software capabilities on which mobile services users have come to rely, such as maps and voice recognition. This movement of processing cycles and storage away from users' machines to data centers is, however, entirely dependent upon the availability of broadband. The more data intensive the service, the more bandwidth that is required. For many classes of applications, such as video editing, bandwidth requirements and availability continue to make cloud-based processing an unattractive option.[13] Indeed, data-intensive services such as Second Life or Google Earth must be accessed through client software (or web plug-ins) that can render 3-D virtual environments with the help of the device's local processor. Furthermore, in the cloud model, reliability is entirely a function of service providers, themselves dependent upon network service providers. We might say, "live by the Cloud, die by the network."

Already, the centralization brought on by data centers is being mitigated by content distribution networks (CDNs) such as Akamai, designed to move resources (processing, storage, and data) closer to the edge of the networks—that is, users' machines. Large content providers like Netflix are developing their own proprietary CDNs (for example, OpenConnect) to bypass the costs and network congestion that result from moving large amounts of data across the Internet. And of course, data centers are themselves being strategically located so as to minimize not only data movement but also energy costs. Even in a cloud-based world, then, there remain multiple models for the distribution of computing resources, whether by reason of computing efficiency or political conviction, and spatial distribution of these resources continues to matter. The Cloud thus emerges at the historical confluence of several longstanding technical traditions within computing: modularity, which has allowed cloud providers to create unprecedented amounts of computing power by merely bringing together massive numbers of low-cost off-the-shelf components; virtualization, which makes it possible to distribute, meter, and charge for these computing resources in highly granular and flexible ways while allowing for continuity with legacy software designs; and distributed architectures, which allow for the partitioning of computing resources between mobile devices and data centers.

The Era of Computational Scarcity

Digital exceptionalism thus holds limited analytical power to illuminate the dynamics that have resulted in today's particular infrastructural configurations. Conversely, a focus on the material dimension of bits provides an explanatory framework that does much for understanding the debates that currently animate the world of computing. Perhaps the most important of these debates (and the one that stands in starkest contrast to the vision of digital exceptionalism) concerns the supply and demand of computational resources. The demand for these resources is virtually inexhaustible because there are, in effect, no particular limits on the amounts of digital information that can be created. Indeed, digital images, films, and sounds are constantly gaining in resolution. Internet traffic increased eightfold between 2006 and 2011 and is expected to continue growing at an annual rate of 30 percent. Mobile traffic now accounts for 45 percent of all IP traffic.[14] There is little end in sight for such demand. The project of computing as it stands today is to provide no less than a computational mirror of the whole world, a global simulation that would augment its physical counterpart so as to make it more predictable and more amenable to manipulation.[15] The resolution of this simulation increases with each new sensor developed, deployed, and plugged into the network, with each

new data point integrated into its statistical machinery. The consequence is that the rate of growth of digital data will in all likelihood continue to exceed the available material resources to manipulate, store, and move about this data for the foreseeable future.

A more useful framework, then, for understanding the evolution of computing is one of computational scarcity—that is, of differential access to these limited computing resources by different groups, whether the result of geography, socioeconomic status, legal status, or other causes. These differentials will manifest themselves in familiar ways—individuals will or will not have access to computing devices, bandwidth, or data—but also in new and unfamiliar ways, such as through the design and deployment of largely invisible infrastructural elements that will enable or disable certain kinds of sharing and distribution of computational resources. Indeed, our era's true claim to exceptionalism will likely lay in the subtlety and reach of the information inequalities caused by the computing infrastructure.

01. This article draws in part from Jean-François Blanchette, "Computing's Infrastructural Moment," in *Regulating the Cloud: Policy for Computing Infrastructure*, eds. Christopher S. Yoo and Jean-François Blanchette (Cambridge, MA: MIT Press, 2015).
02. Nicholas Negroponte, *Being Digital* (New York: Random House, 1996).
03. Lawrence Lessig, "The Architecture of Innovation," *Duke Law Journal* 51 (2002): 1783–1801; Jonathan Zittrain, *The Future of the Internet—and How to Stop It* (New Haven, CT: Yale University Press, 2008); Tim Wu, *The Master Switch: The Rise and Fall of Information Empires* (New York: Alfred A. Knopf, 2010).
04. See James Gleick, *The Information: A History, A Theory, A Flood* (New York: Pantheon Books, 2011); and George Dyson, *Turing's Cathedral: The Origins of the Digital Universe* (New York: Pantheon Books, 2012).
05. This is the doctrine of binarism, an ancillary principle to digital exceptionalism. For more on this, see Mathieu Triclot, *Le Moment Cybernétique: La Constitution De La Notion d'Information* (Seyssel: Champ Vallon, 2008).
06. Indeed, Turing himself viewed the physics of computation as wholly subordinate to its mathematics, as he noted in a speech in 1947: "From the point of view of the mathematician the property of [an electronic digital computing machine] being digital should be of greater interest than that of being electronic. That it is electronic is certainly important because these machines owe their high speed to this, and without the speed it is doubtful if financial support for their construction would be forthcoming. But this is virtually all there is to be said on that subject." Alan Turing, "Lecture on the Automatic Computing Engine (1947)," in *The Essential Turing: Seminal Writings in Computing, Logic, Philosophy, Artificial Intelligence, and Artificial Life: Plus the Secrets of Enigma*, ed. B. Jack Copeland (New York: Oxford University Press, 2004), 362.
07. See A. K. Dewdney, "Computer Recreations: A Tinkertoy Computer That Plays Tic-Tac-Toe," *Scientific American* 261, no. 4 (October 1989): 119–23; and D. Waitzman, "A Standard for the Transmission of IP Datagrams on Avian Carriers," *IETF RFC* 1149 (April 1, 1990).
08. In Jean-François Blanchette, "Computing as if Infrastructure Mattered," *Communications of the ACM* 55, no. 10 (October 2012): 32–34, I argue in favor of a focus on infrastructural forces and that the material foundation of computing resources provides an appropriate pedagogical framework for teaching the historical evolution of computing over the current emphasis on mathematical abstraction.
09. For example: "The unprecedented evolution of computers since 1980 exhibits an essentially exponential speedup that spans 4 orders of magnitude in performance for the same (or lower) price. No other engineered system in human history has ever achieved that rate of improvement, […] Whole fields of human endeavors have been transformed as computer system capability has ascended through various threshold performance values." Lynette I. Millett and Samuel H. Fuller, eds., *The Future of Computing Performance: Game Over or Next Level?* (Washington, DC: National Academies Press, 2011), 25.
10. See Jean-François Blanchette, "A Material History of Bits," *Journal of the American Society for Information Science and Technology* 62, no. 6 (June 2011): 1042–57.
11. *Principles of Operation—Type 701 and Associated Equipment* (New York: International Business Machines Corporation [IBM], 1953).
12. Carliss Young Baldwin and Kim B. Clark, *Design Rules*, vol. 1, *The Power of Modularity* (Cambridge, MA: MIT Press, 2000).
13. Ironically, in many cases, the transfer of datasets to the Cloud can be cost prohibitive leading Armbrust et al. to recommend the "FedEx disk option," that is, ship them using a more traditional infrastructure. See M. Armbrust et al., *Above the Clouds: A Berkeley View of Cloud Computing*, Technical Report UCB/EECS-2009-28 (Berkeley CA: EECS Department, University of California, Berkeley, 2009).
14. See *Cisco Visual Networking Index: Global Mobile Data Traffic Forecast Update, 2012–2017*, http://newsroom.cisco.com/documents/10157/1142732/Cisco_VNI_Mobile_Data_Traffic_Forecast_2012_2017_white_paper.pdf.
15. See, for example, David Hillel Gelernter, *Mirror Worlds, or, the Day Software Puts the Universe in a Shoebox: How It Will Happen and What It Will Mean* (New York: Oxford University Press, 1991).

Image Credits

022, 024: Photos by Shuli Hallak.