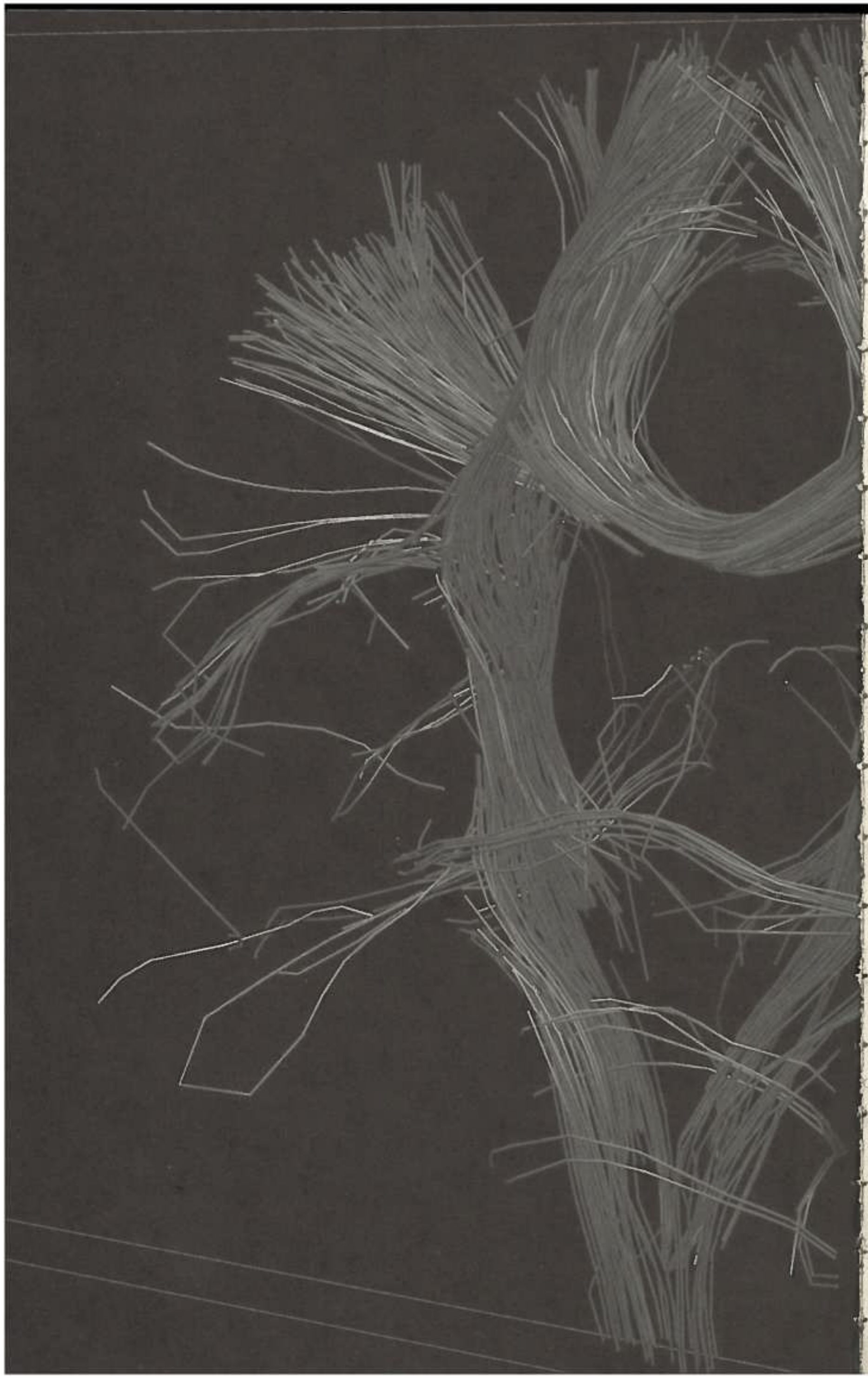

JOHN
VON
NEUMANN

THE COMPUTER & THE BRAIN

FOREWORD BY RAY KURZWEIL

THIRD EDITION

Yale
UNIVERSITY
PRESS
NEW HAVEN
& LONDON

CONTENTS

FOREWORD TO THE THIRD EDITION

Information technologies have already transformed every facet of human life from business and politics to the arts. Given the inherent exponential increase in the price-performance and capacity of every form of information technology, the information age is continually expanding its sphere of influence. Arguably the most important information process to understand is human intelligence itself, and this book is perhaps the earliest serious examination of the relationship between our thinking and the computer, from the mathematician who formulated the fundamental architecture of the computer era.

In a grand project to understand the human brain, we are making accelerating gains in reverse engineering the paradigms of human thinking, and are applying these biologically inspired methods to create increasingly intelligent machines. Artificial intelligence (AI) devised in this way will ultimately soar past unenhanced human thinking. My view is that the purpose of this endeavor is not to displace us but to expand the reach of what is already a human-machine civilization. This is what makes our species unique.

So what are the key ideas that underlie this information age? By my count there are five. John von Neumann is largely responsible for three of them, and he made a fundamental contribution to a fourth. Claude Shannon solved the fundamental problem of making information reliable. Alan Turing demonstrated and defined the universality of computation and was influenced by an early lecture by von Neumann. Building on Turing and Shannon, von Neumann created the von Neumann machine, which became—and remains—the fundamental architecture for computation.

In the deceptively modest volume you are now holding, von Neumann articulates his model of computation and goes on to define the essential equivalence of the human brain and a computer. He acknowledges the apparently deep structural differences, but by applying Turing's principle of the equivalence of all computation, von Neumann envisions a strategy to understand the brain's methods as computation, to re-create those methods, and ultimately to expand its powers. The book is all the more prescient given that it was written more than half a century ago when neuroscience had only the most primitive tools available. Finally, von Neumann anticipates the essential acceleration of technology and its inevitable consequences in a coming singular transformation of human existence. Let's consider these five basic ideas in slightly more detail.

Around 1940, if you used the word "computer," people assumed you were talking about an analog computer. Numbers were represented by different levels of voltage, and specialized components could perform arithmetic functions such as addition and multiplication. A big limitation, however, was that analog computers were plagued by accuracy issues. Numbers could be represented with an accuracy of only about one part in a hundred, and because voltage levels representing numbers were processed by increasing numbers of arithmetic operators, these errors would accumulate. If you wanted to perform more than a handful of computations, the results would become so inaccurate as to be meaningless.

Anyone who can remember the days of copying music using analog tape will remember this effect. There was noticeable degradation on the first copy, for it was a little noisier than the original ("noise" represents random inaccuracies). A copy of the copy was noisier still, and by the tenth generation, the copy was almost entirely noise.

It was assumed that the same problem would plague the emerging world of digital computers. We can see this perceived problem if we consider the communication of digital information through a channel. No channel is perfect and will have some inherent error rate. Suppose we have a channel that has a 0.9 probability of correctly transmitting each bit of information. If I send a message that is one-bit long, the probability of accurately transmitting it through that channel will be 0.9. Suppose I send two bits? Now the accuracy is $0.9^2 = 0.81$. How about if I send one byte (eight bits)? I have less than an even chance (0.43, to be exact) of sending it correctly. The probability of accurately sending five bytes is about 1 percent.

An obvious approach to circumvent this problem is to make the channel more accurate. Suppose the channel makes only one error in a million bits. If I send a file with a half million bytes (about the size of a modest program or database), the probability of correctly transmitting it is less than 2 percent, despite the very high inherent accuracy of the channel. Given that a single-bit error can completely invalidate a computer program and other forms of digital data, that is not a satisfactory situation. Regardless of the accuracy of the channel, since the likelihood of an error in a transmission grows rapidly with the size of the message, this would seem to be an intractable problem.

Analog computers approach this problem through graceful degradation. They also accumulate inaccuracies with increased use, but if we limit ourselves to a constrained set of calculations, they prove somewhat useful. Digital computers, on the other hand, require continual communication, not just from one computer to another, but within the computer itself. There is communication between its memory and the central processing unit. Within the central processing unit, there is communication from one register to another, and back and forth to the arithmetic unit, and so

on. Even within the arithmetic unit, there is communication from one bit register to another. Communication is pervasive at every level. If we consider that error rates escalate rapidly with increased communication and that a single-bit error can destroy the integrity of a process, digital computation is doomed—or so it seemed at the time.

Remarkably, that was the common view until Shannon came along with the first key idea of the information age. He demonstrated how we can create arbitrarily accurate communication using the most unreliable communication channels. What Shannon said in his now landmark paper, "A Mathematical Theory of Communication," published in the *Bell System Technical Journal* in July and October 1948, and in particular in his noisy-channel coding theorem, was that if you have available a channel with any error rate (except for exactly 50 percent per bit, which would mean that channel is transmitting pure noise), you are able to transmit a message and make the error rate as accurate as you want. In other words, the error rate can be one bit out of *n* bits, where *n* can be as large as you want. So, for example, in the extreme, if you have a channel that transmits bits of information correctly only 51 percent of the time (that is, it transmits the correct bit just slightly more often than the wrong bit), you can nonetheless transmit messages such that only one bit out of a million is incorrect, or one bit out of a trillion or a trillion trillion.

How is this possible? The answer is through redundancy.

That may seem obvious now, but it was not obvious at the time. As a simple example, if I transmit each bit three times and take the majority vote, I will have substantially increased the reliability of the result. If that is not good enough, increase the redundancy until you get the reliability you need. The idea of simply repeating information is the easiest way to see how we can achieve arbitrarily high accuracy rates from low-accuracy channels, but this approach is not the most efficient approach. Shannon's paper established the field of information theory and presented optimal methods of error detection and correction codes that can achieve any target accuracy through any nonrandom channel.

Older readers will recall telephone modems that transmitted information through noisy analog phone lines, which included audible hisses and pops and many other forms of distortion, but nonetheless were able to transmit digital data with very high accuracy rates, thanks to Shannon's noisy-channel theorem.

The same issue and the same solution exist for digital memory. Ever wonder how CDs, DVDs, and program disks continue to provide reliable results even after the disk has been dropped on the floor and scratched? Again, we can thank Shannon. Computation consists of three elements: communication (which, as I mentioned, is pervasive both within and between computers), memory, and logic gates (which perform the arithmetic and logical functions). The accuracy of logic gates can also be made arbitrarily high by

similarly using error detection and correction codes. It is due to Shannon's theorem that we can handle arbitrarily large and complex digital data and algorithms without the processes being disturbed or destroyed by errors.

The second important idea on which the information age relies is the universality of computation. In 1936 Alan Turing described his "Turing machine," which is not an actual machine but a thought experiment. His theoretical computer consists of an infinitely long memory tape with a 1 or a 0 in each square. Input to the machine is presented on this tape. The machine reads the tape one square at a time. The machine also contains a table of rules, essentially a stored program. The rules consist of numbered states. Each rule specifies one action if the square currently being read is a 0 and a different action if the current square is a 1. Possible actions include writing a 0 or a 1 on the tape, moving the tape one square to the right or the left, or halting. Each state then specifies the number of the next state that the machine should be in. When the machine halts, it has completed its algorithm, and the output of the process is left on the tape. Even though the tape is theoretically infinite in length, any actual program (that does not get into an infinite loop) uses only a finite portion of the tape, so if we limit ourselves to a finite memory, the machine still solves a useful set of problems.

If the Turing machine sounds simple, that was Turing's objective. He wanted his Turing machine to be as simple as possible (but no simpler, to paraphrase Einstein). Turing and Alonzo Church, his former professor, went on to develop the Church-Turing thesis, which states that if a problem that can be presented to a Turing machine is not solvable by a Turing machine, it is also not solvable by *any* machine, following natural law. Even though the Turing machine has only a handful of commands and processes only one bit at a time, it can compute anything that any computer can compute.

"Strong" interpretations of the Church-Turing thesis propose an essential equivalence between what a human can think or know and what is computable by a machine. The basic idea is that the human brain is subject to natural law, and thus its information-processing ability cannot exceed that of a machine (and therefore of a Turing machine).

We can properly credit Turing with establishing the theoretical foundation of computation with his 1936 paper, but it is important to note that Turing was deeply influenced by a lecture that John von Neumann gave in Cambridge, England, in 1935 on his stored-program concept, a concept enshrined in the Turing machine. In turn, von Neumann was influenced by Turing's 1936 paper, which elegantly laid out the principles of computation, and he made it required reading for his colleagues in the late 1930s and early 1940s.

In the same paper, Turing reports another unexpected discovery, that of unsolvable problems. These are problems that are well defined and have unique answers that can be shown to exist but that we can also prove can never be computed by

a Turing machine—that is to say, by *any* machine—which is a reversal of what had been a nineteenth-century confidence that problems that could be defined would ultimately be solved. Turing showed that there are as many unsolvable problems as solvable ones. Kurt Gödel reached a similar conclusion in his 1931 "Incompleteness Theorem." We are thus left with the perplexing situation of being able to define a problem, to prove that a unique answer exists, and yet to know that the answer can never be discovered.

A lot more can be said about the philosophical implications of the work of Turing, Church, and Gödel, but for the purposes of this foreword, it is sufficient to say that Turing showed that computation is essentially based on a very simple mechanism. Because the Turing machine (and, therefore, any computer) is capable of basing its future course of action on results it has already computed, it is capable of making decisions and modeling arbitrarily complex hierarchies of information.

Turing designed and completed what is arguably the first computer, called the Colossus, by December 1943 to decode messages that had been encrypted by the Nazi Enigma coding machine. It was designed for one task and could not be reprogrammed for a different task. But it performed this one task brilliantly and is credited with enabling the Allies to overcome the 3:1 advantage that the German Luftwaffe enjoyed over the British Royal Air Force, enabling the Allies to win the crucial Battle of Britain.

It was on these foundations that John von Neumann created the architecture of the modern computer, the von Neumann machine, which has remained the core structure of essentially every computer for the past sixty-six years, from the microcontroller in your washing machine to the largest supercomputers. This is the third key idea of the information age. In a paper dated June 30, 1945, titled "First Draft of a Report on the EDVAC," von Neumann presented the concepts that have dominated computation ever since. The von Neumann model includes a central processing unit where arithmetical and logical operations are carried out, a memory unit where the program and data are stored, mass storage, a program counter, and input/output channels. This conception is described in the first half of this book. Although von Neumann's paper was intended as an internal project document, it became the Bible for computer designers in the 1940s and 1950s and, indeed, has influenced the building of every computer since that time.

The Turing machine was not designed to be practical. Turing's theorems were not concerned with the efficiency of solving problems but rather with examining the range of problems that could be solved by computation. Von Neumann's goal was to create a practical concept of a computational machine. His concept replaces Turing's one-bit computations with multiple-bit words (generally some multiple of eight bits). Turing's memory tape is sequential, so Turing machine programs spend an inordinate amount of time

moving the tape back and forth to store and retrieve intermediate results. In contrast, von Neumann's machine has a random access memory, so any data item can be immediately retrieved.

One of von Neumann's key concepts is the stored program, which he had introduced a decade earlier: the program is stored in the same type of random access memory as the data (and often in the same block of memory). This allows the computer to be reprogrammed for different tasks. It even allows for self-modifying code (if the program store is writable), which allows for a powerful form of recursion. Up until that time, virtually all computers, including Turing's own Colossus, were built for a specific task. The stored program allows a computer to be truly universal, thereby fulfilling Turing's vision of the universality of computation.

Another of von Neumann's key concepts is for each instruction to include an operation code specifying the arithmetic or logical operation to be performed and the address of an operand from memory. Von Neumann's formulation was introduced with his publication of the design of the EDVAC, a project he conducted with collaborators J. Presper Eckert and John Mauchly. The EDVAC itself did not actually run until 1951, by which time there were other stored-program computers, such as the Manchester Small-Scale Experimental Machine, ENIAC, EDSAC, and BINAC, all of which had been deeply influenced by von Neumann's paper and involved Eckert and Mauchly as designers. Von Neumann was a direct contributor to the design of a number of these machines, including a later version of ENIAC which supported a stored program.

There were a few precursors to von Neumann's architecture, although none were true von Neumann machines, with one surprising exception. Howard Aiken's Mark I, built in 1944, had an element of programmability but did not use a stored program. It read instructions from a punched paper tape and then executed each command immediately. There was no conditional branch instruction, so it cannot be considered to be an example of von Neumann architecture.

Predating the Mark I was the creation of the Z-3 computer in 1941 by Konrad Zuse. It also read its program from a tape (in this case, coded on film) and also lacked a conditional branch instruction. Interestingly, Zuse had support from the German Aircraft Research Institute which used the Z-3 to study wing flutter, but his proposal to the Nazi government for funding to replace his relays with vacuum tubes was turned down. The Nazis deemed computation as "not war important."

The one true precursor to von Neumann's concept came a full century earlier. Charles Babbage's Analytical Engine, which he first described in 1837, incorporated the idea of a stored program, which it provided via punched cards borrowed from the Jacquard loom. Its random access memory included one thousand words of 50 decimal digits each

(the equivalent of about 21 kilobytes). Each instruction included an operation code and an operand number, just like modern machine languages. It also included conditional branching and looping, so it was a true von Neumann machine. It appears that the Analytical Engine was beyond Babbage's mechanical and organizational skills, and it never ran. It is unclear whether the twentieth-century pioneers of the computer, including von Neumann, were aware of Babbage's work.

Despite never running, Babbage's computer resulted in the creation of the field of software programming. Ada Byron, Countess of Lovelace and the only legitimate child of the poet Lord Byron, wrote programs for the Analytical Engine that she needed to debug in her own mind, a practice well known to software engineers today as "table checking." She translated an article by the Italian mathematician Luigi Menabrea on the Analytical Engine and added extensive notes of her own. She wrote that "the Analytical Engine weaves algebraic patterns, just as the Jacquard loom weaves flowers and leaves." She went on to provide perhaps the first speculations on the feasibility of artificial intelligence, but concluded that the Analytical Engine had "no pretensions whatever to originate anything."

Babbage's conception is quite miraculous when you consider the era in which he lived and worked. However, by the mid-twentieth century, his work had been lost in the mists of time. It was von Neumann who conceptualized and articulated the key principles of the computer as we know it today, and the world recognizes this by continuing to refer to the von Neumann machine as the principal model of computation.

Keep in mind that the von Neumann machine continually communicates data between and within its various units, so it would not be possible to build one if it were not for Shannon's theorems and the methods he devised for transmitting and storing reliable digital information.

That brings us to the fourth important idea, which is to find ways to endow computers with intelligence, to go beyond Ada Byron's conclusion that a computer cannot think creatively. Alan Turing had already introduced this goal with his 1950 paper, "Computing Machinery and Intelligence," which includes his now famous "Turing test" for ascertaining whether or not an AI has achieved a human level of intelligence. In this book, after introducing von Neumann architecture, von Neumann looks to the human brain itself. The human brain is, after all, the best example we have of an intelligent system. If we can learn its methods, we can use these biologically inspired paradigms to build more intelligent machines. This book is the earliest serious examination of the human brain from the perspective of a mathematician and computer pioneer. Prior to von Neumann, the fields of computer science and neuroscience were two islands with no bridge between them.

It is ironic that the last work of one of the most brilliant

mathematicians of the twentieth century and one of the pioneers of the computer age was an examination of intelligence itself. The work was intended as a series of lectures for Yale University, but because of the ravages of cancer, von Neumann never delivered the lectures and did not complete the manuscript from which the lectures were to be given. It nonetheless remains a brilliant and prophetic foreshadowing of what I regard as humanity's most daunting and important project.

Von Neumann starts by articulating the differences and similarities between the computer and the human brain. Given that he wrote in 1955 and 1956, the manuscript is remarkably accurate, especially in the details that are pertinent to the comparison. He notes that the output of neurons is digital: an axon either fires or it doesn't. This was far from obvious at the time, in that the output could have been an analog signal. The processing in the dendrites leading into a neuron and in the soma neuron cell body, however, are analog. He describes these calculations as a weighted sum of inputs with a threshold. This model of how neurons work led to the field of connectionism, in which systems are built based on this neuron model in both hardware and software. The first such connectionist system was created by Frank Rosenblatt as a software program on an IBM 704 computer at Cornell in 1957.

We now have more sophisticated models of how neurons combine inputs, but the essential idea of analog processing of dendrite inputs using neurotransmitter concentrations has held up. We would not have expected von Neumann to get all of the details of how neurons process information correct in 1956, but the key points on which he bases his arguments remain valid.

Von Neumann applies the concept of the universality of computation to conclude that even though the architecture and building blocks of the brain and the computer appear to be radically different, we can nonetheless conclude that a von Neumann machine can simulate a brain's processing. The converse does not hold, however, because the brain is not a von Neumann machine and does not have a stored program as such. Its algorithm, or methods, are implicit in its structure.

Von Neumann correctly concludes that neurons can learn patterns from their inputs, which we now know are coded in neurotransmitter concentrations. What was not known in von Neumann's time is that learning also takes place through the creation and destruction of connections between neurons.

Von Neumann notes that the speed of neural processing is extremely slow, on the order of a hundred calculations per second, but that the brain compensates for this through massive parallel processing. Each one of its 10^{10} neurons is processing simultaneously (this number is also reasonably accurate; estimates today are between 10^{10} and 10^{11}). In fact, each of the connections (with an average of about 10^3 connections per neuron) is computing simultaneously.

His estimates and description of neural processing are remarkable, given the primitive state of neuroscience at the time. One description that I disagree with is von Neumann's estimate of the brain's memory capacity. He assumes that the brain remembers every input for an entire lifetime. Sixty years is about 2×10^9 seconds. With about fourteen inputs to each neuron per second (which is actually low by at least three orders of magnitude), and with 10^{10} neurons, he gets an estimate of about 10^{20} bits for the brain's memory capacity. The reality is that we remember only a very small fraction of our thoughts and experiences, and these memories are not stored as bit patterns at a low level (such as a video image), but rather as sequences of higher-level patterns. Our cortex is organized as a hierarchy of pattern recognizers. Some of these recognizers will recognize certain topological forms, such as the cross bar in a capital "A" or its lower concavity. These low-level recognizers in the neocortex feed those recognitions to pattern recognizers at a higher level. At that next level, recognizers may recognize specific printed letters such as the letter "A." At an even higher level, words may be recognized, such as the word "Apple." In another portion of the cortex, a recognizer at a comparable level may recognize the object, an apple; and in yet another portion, a recognizer may recognize the spoken word, "apple." At a much higher conceptual level, a recognizer may conclude, "That was funny." Our memory of events and thoughts is coded in terms of these higher-level recognitions. If we recall a memory of an experience, there is nothing equivalent to a video playing in our head. Rather, we recall a sequence of these high-level patterns. We have to reimagine the experience, for the details are not explicitly remembered.

You can demonstrate this to yourself by trying to recall a recent experience—for example, the last time you took a walk. How much of that experience do you remember? Who was the fifth person you encountered? Did you see a baby carriage? A mailbox? What did you see when you turned the first corner? If you passed some stores, what was in the second window? Perhaps you can reconstruct the answers to these questions from the few cues that you remember, but most of us do not have perfect recall of our experiences. Machines can, in fact, recall easily, and that is one advantage of artificial intelligence.

There are very few discussions in this book that I find to be at significant odds with what we now understand. We are not in a position today to describe the brain perfectly, so we would not expect a book from 1956 on reverse engineering the brain to do so. That being said, von Neumann's descriptions are remarkably up to date, and the details on which he bases his conclusions remain valid. As he describes each mechanism in the brain, he shows how a modern computer could accomplish the same operation, despite the apparent differences. The brain's analog mechanisms can be simulated through digital ones because digital computation can emulate analog

values to any desired degree of precision (and the precision of analog information in the brain is quite low).

The brain's massive parallelism can also be simulated, given the significant speed advantage of computers in serial computation (an advantage that has vastly expanded since the book was written). In addition, we can also use parallel processing in computers by using parallel von Neumann machines. That is exactly how supercomputers work today.

Considering how quickly we are able to make decisions and how very slowly neurons compute, he concludes that the brain's methods cannot involve lengthy sequential algorithms. When a baseball fielder on third base decides to throw to first rather than to second base, he makes this decision in a fraction of a second. There is time for each neuron to go through only a handful of cycles (the period of time necessary for neural circuits to consider new inputs). Von Neumann correctly concludes that the brain's remarkable powers come from the ten billion neurons being able to process information all at the same time. Recent advances in reverse engineering the visual cortex have confirmed that we make sophisticated visual judgments in only three or four neural cycles.

There is considerable plasticity in the brain, which enables us to learn. But there is far greater plasticity in a computer, which can completely restructure its methods by changing its software. Thus a computer will be able to emulate the brain, but the converse is not the case.

When von Neumann compared the capacity of the brain's massively parallel organization to the (few) computers of his time, it was clear that the brain had far greater capacity than computers circa 1956. Today the first supercomputers are being built that achieve a speed matching some of the more conservative estimates of the speed required to functionally simulate the human brain (about 10^{16} operations per second). I estimate that the hardware for this level of computation will cost $1,000 early in the 2020s. Even though it was remarkably early in the history of the computer when this manuscript was written, von Neumann nonetheless had confidence that both the hardware and the software of human intelligence would ultimately be available. That was the reason he prepared these lectures.

Von Neumann was deeply aware of the accelerating pace of progress and the profound implications of this progression for humanity's future, which brings us to the fifth key idea of the information age. A year after von Neumann's death in 1957, fellow mathematician Stan Ulam quoted von Neumann as having said that "the ever accelerating progress of technology and changes in the mode of human life give the appearance of approaching some essential singularity in the history of the race beyond which human affairs, as we know them, could not continue." This is the first known use of the word "singularity" in the context of human history.

Ray Kurzweil

Further Readings

Bochner, S. (1958). *A Biographical Memoir of John von Neumann*. Washington, D.C.: National Academy of Sciences.

Turing, A. M. (1936). "On Computable Numbers, with an Application to the Entscheidungsproblem." *Proceedings of the London Mathematical Society* 42: 230–65. doi:10.1112/plms/s2-42.1.230.

Turing, A. M. (1938). "On Computable Numbers, with an Application to the Entscheidungsproblem: A Correction." *Proceedings of the London Mathematical Society* 43: 544–46. doi:10.1112/plms/s2-43.6.544.

Turing, A. M. (October 1950). "Computing Machinery and Intelligence." *Mind* 59 (236): 433–60.

Ulam, S. (May 1958). "Tribute to John von Neumann." *Bulletin of the American Mathematical Society* 64 (3, part 2): 1–49.

von Neumann, John. (June 30, 1945). "First Draft of a Report on the EDVAC." Moore School of Electrical Engineering, University of Pennsylvania.

von Neumann, John. (July and October 1948). "A Mathematical Theory of Communication." *Bell System Technical Journal* 27.