DOI:10.1145/2629499

Edgar G. Daylight

# Viewpoint
# A Turing Tale

*Assessing the accuracy of popular descriptions of Alan Turing's influences and legacy.*

PHOTOGRAPH BY THOMAS BETHGE

Much has been written about Alan Turing during the past decades and by a variety of people, including historians, philosophers, and logicians. Becoming a Turing scholar today not only requires archival research but also the study of several secondary sources. Doing the latter leads to the observation that many texts contain flaws.

In this Viewpoint, I compare and contrast some key arguments put forth by three Turing scholars—Andrew Hodges, Martin Davis, and Jack Copeland—highlighting the conceptual difference between a "universal Turing machine" and a "stored program" computer. My findings complement Thomas Haigh's January 2014 *Communications* Historical Reflections column, "Actually, Turing Did Not Invent the Computer."[7]

In his 1936 paper, "On Computable Numbers," Turing introduced his automatic machines, which do not contain a finite output (nor an input) as is the case with the later-devised "Turing machines." Turing wanted each of his machines to compute and print a real number (such as $\pi$ and $\frac{1}{4}$). For example, the machine computing $\frac{1}{4}$ prints the digits 0 and 1 and then forever prints the digit 0 in accordance with $\frac{1}{4}$'s binary representation: 0.01000...

During the course of three decades, Turing, Emil Post, Alonzo Church, Stephen Kleene, Martin Davis, Saul Gorn, and others recast the concept of Turing's 1936 automatic machine. Several years were needed for the term "universal Turing machine" to acquire an invariant meaning.[5,12] Martin Davis presented a modern definition in his 1958 book *Computability and Unsolvability*[3] and a definition for the layman in his recent book *The Universal Computer: The Road from Leibniz to Turing*[4]—two definitions I abide with here and with which modern textbooks in computer science comply.

The meaning attached to the words "stored program" also changed in the post-war years and it is unlikely those words meant exactly the same to every historical actor. In this Viewpoint, they refer to a large store inside the computer, containing both numbers and instructions. According to the current state of the art in the history of computing, the words "stored program" were introduced in 1949 by IBM engineers in Poughkeepsie, NY.[8]

Although all three Turing scholars have their own unique narrative thrust, I will discuss Hodges's 1983 biography first and then scrutinize Davis's and Copeland's work together. Davis and Copeland have more in common than meets the eye.

## Hodges

In his authoritative biography, Hodges put Turing's life in a pluralistic context, as the following points illustrate:

- Also in a world without Turing, his universal machine would have come to light in one form or another and in no small part due to Emil Post, even though Post's "worker" model did not include a "universal machine" construction.[9]
- One hundred years before Turing, Babbage had already planned for storing numbers in a machine that was universal, as he and Ada (Countess of Lovelace) were well aware.[9]
- In America, Eckert and Mauchly perceived the idea of storing instructions inside the machine, in electronic form.[9]
- Von Neumann may or may not have been influenced by Turing when working on the ENIAC-to-EDVAC transition.[9]

Hodges stressed throughout his book that Turing was not taken seriously by most of his contemporaries in the arena of computer building.[9] Turing's 1936 paper meant a lot to him and to some of his close colleagues, as Charles G. Darwin's repeated statements in 1946 about the ACE machine illustrate.[9] That said, Turing's paper had little impact on the computer-building community at large.[9]

In what respect, then, did Turing stand out in the 1940s?

Turing had the remarkable ability to unify seemingly disparate theoretical *and* practical concepts. He needed just one tape in his 1936 paper and just one electronic memory in the 1940s. In Hodges's words: "This [unification of data and instructions] was the new idea, ... For it threw all the emphasis on to a new place—the construction of a large, fast, effective, all-purpose electronic 'memory.' And in its way it made everything much more simple, less cluttered in conception."[9]

The idea to unify was, however, not solely Turing's, nor did it require knowledge of Turing's 1936 paper per se. But, Turing's unification was, unlike that of most of his contemporaries, also theoretical in nature. Based on his 1936 universal machine, Turing was able to see that one machine could do the job of several special-purpose machines.[9] This grander picture of computing was something Turing was not able to convey clearly in the 1940s to his contemporaries who were eagerly, and successfully, building modern computers. That, in brief, is what I take to be Hodges's central technical theme.

**Turing had the remarkable ability to unify seemingly disparate theoretical *and* practical concepts.**

Hodges distinguished between Babbage's universal physical machine and Turing's universal physical machine. Babbage had not stored instructions internally in his machine while Turing planned to do just that. As Hodges implicitly conveyed, storing instructions externally (on, say, paper tape) or internally (in computer memory) does not matter in terms of Turing's universality.[9] Men like Turing and von Neumann very likely understood this theoretical connection during the 1940s, while many contemporaries did not.[a]

## Davis and Copeland

In contrast to Hodges, both Davis and Copeland depict history as a stream, as a Turing Tale, starting with Turing's abstract 1936 paper and ending with the modern computer.[2] They completely neglect that grasping the emerging practical implications of Turing's 1936 paper took several years, even for logicians.

To set the stage for Turing, Davis refers to a tiny excerpt from Babbage's writings—which states that his analytical engine "could do everything but compose country dances"—to conclude that Babbage had a limited view on universality.[4]

Davis also puts Turing on a pedestal by ridiculing the following 1955 statement of the computer pioneer Howard Aiken: "If it should turn out that the basic logics of a machine designed for the numerical solution of differential equations coincide with the logics of a machine intended to make bills for a department store, I would regard this as the most amazing coincidence that I have ever encountered."[4]

Davis rightfully observes that Aiken did not grasp Turing's notion of universality. But Davis takes one step too many when he portrays Aiken as someone who was therefore lagging behind on the current events of his time. In Davis's words: "If Aiken had grasped the significance of Alan Turing's paper, published two decades earlier, he would never have made such a preposterous statement."[4]

A more careful interpretation of history, by contrast, characterizes Aiken as one of several computer professionals who simply did not have to rely on Turing's "universal machine" concept to advance computer technology—an observation that is apparently difficult to make by an eminent logician who has experienced the past differently. Davis was in fact still writing down his newly acquired insights between Turing's theory and computing practice in his book *Computability and Unsolvability*[b] when Aiken made his "preposterous" statement in 1955. Davis's 1958 book opened the eyes of many mathematically inclined computer programmers,[11] a large percentage of which had not heard of Turing until then.

Davis also notes that "Turing's universal computer was a marvel-

a I am giving Turing and von Neumann the benefit of the doubt here. This is similar to Hodges's conjecture that Turing also knew all along that "program modification does not extend the range of possible operations" either.[9] Hodges noted Herman Goldstine apparently did not see this connection, thereby suggesting Turing's 1936 paper was received in different (and less thorough) ways among those who did come across his work. (Goldstine had eagerly read Turing's 1936 paper by January 8, 1946.[6] My thanks to Thomas Haigh and Mark Priestley for bringing Herman Goldstine's letter to my attention.) I opine that many people today mistakenly take "program modification" to be part and parcel of Turing's 1936 universal-machine construction.

b As Davis confirmed to me (Ghent, November 8, 2011).

ous conceptual device," which is of course true, and then continues as follows: "But could one actually build such a thing?...These questions were in Turing's mind from the very first."[4] Hodges, by contrast, has cautioned his readership not to blindly believe that Turing set about constructing a universal machine before the war.[9] Although Turing was a very creative and rather unusual mathematician, he, too, needed time to connect his 1936 work on logic to rapidly changing technology.

Copeland goes even further than Davis by conflating the "universal machine" and the "stored program" concepts. (Copeland is not alone. See, for example, Priestley's scrutiny of Ceruzzi's work.[13]) Copeland misleadingly describes "the stored-program universal computer" as a "single invention" from 1936[2]—a statement that both Hodges and Davis have complained about.[4,10]

"Computer science textbooks," Copeland says, "often attribute Turing's stored-program concept to von Neumann." But, Copeland insists, von Neumann "never claimed it as his own."[2] Copeland then continues: "On the contrary, von Neumann said that Turing's 'great positive contribution' was to show that 'one, definite mechanism can be *universal*.'"[c,2]

Von Neumann referred here to Turing's 1936 universal machine, which is not the same as the "stored program" computer of the 1940s. A "stored program" is only a means to constructing a practical realization of a "universal Turing machine." Turing and von Neumann were able to make this observation, exactly because they were well versed in both theory and practice. Von Neumann's letter[c] does not support Copeland's reasoning.

Finally, lack of primary sources forces both Davis and Copeland to repeatedly refer to praising, secondhand, comments. Davis relies on *Time* magazine to make a case for Turing.[4] Copeland, in turn, writes: "Many people have acclaimed von Neumann as the 'father of the computer,' von Neumann's friend Stanley Frankel observed, "but I am sure that he would never have made that mistake himself."[2]

c Von Neumann's letter to Wiener, 1946.

**When, how, and why did Turing's work influence other actors during the course of history?**

Here we have one of several references[d] in Copeland's book to contemporaries of von Neumann. Such recollections should be handled with care because they come from people who participated in a success story that was already several decades old and were in no better position than most of us today to identify what only very few men like Turing and von Neumann knew.

## Conclusion

All three Turing scholars—Hodges, Davis, and Copeland—depict the 1940s as a heterogeneous decade. The first half was one of secrecy: Turing was involved in practical issues concerned with electronic computation (albeit for a special kind) and was aware of the completion and significance of the Colossus computers. During the second half of the 1940s, Turing and a small number of by now highly expert colleagues were—despite being unable to tell anyone of the sources of their expertise—actively and publicly involved in computer projects in the U.K.

In retrospect, then, the reader can argue that I have put too much weight on the assessment of Turing's contemporaries. I think most readers will agree with me when I say Turing was a genius. What is relevant here is the following type of question: Did this genius live on a remote island, or did everybody depend on him to advance computer technology? If the former were true, then scholars would hardly be interested in Turing to begin with. Davis and Copeland want us to believe the latter extreme, while Hodges places Turing's life in between both opposites. Historians who want to improve upon Hodge's historiography should ask: When, how, and why did Turing's work influence other actors during the course of history? The assessment of Turing's contemporaries and successors thus matters a great deal.

I conclude that Hodges's 1983 biography is far more accurate than anything that has been written since. From the 1970s onward, popular claims have been made, describing Turing as the "inventor of the computer"; but, see Burks's fitting rebuttal[1] and van Rijsbergen's and Vardi's sober reflections on Turing's legacy[14,15]—a legacy that lies more in programming than in computer building.[5] C

d Frankel's letter to Randell, 1972.

**References**

1. Burks, A.W. The invention of the universal electronic computer—How the electronic computer revolution began. *Future Generation Computer Systems 18* (2002), 871–892.
2. Copeland, B.J. *Turing: Pioneer of the Information Age.* Oxford University Press, 2012.
3. Davis, M. *Computability and Unsolvability.* McGraw-Hill, 1958.
4. Davis, M. *The Universal Computer: The Road from Leibniz to Turing.* CRC Press, 2nd edition, 2012.
5. Daylight, E.G. Towards a historical notion of "Turing—The father of computer science." *History and Philosophy of Logic.* Accepted for publication; see: http://www.dijkstrascry.com/TuringPaper.
6. Goldstine, H.H. untitled (A letter from Goldstine to Womersley, January 8, 1946). Herman H. Goldstine papers in the collection of the American Philosophical Society in Philadelphia, Series 1, Box 3.
7. Haigh, T. Actually, Turing did not invent the computer. *Commun. ACM 57*, 1 (Jan. 2014), 36–41.
8. Haigh, T., Priestley, M., and Rope, C. Reconsidering the stored program concept. *IEEE Annals of the History of Computing* (Jan.–Mar. 2014), 4–17.
9. Hodges, A. *Alan Turing: The Enigma.* Burnett Books, 1983.
10. Hodges, A. Book review: The essential Turing. *Notices of the AMS 53*, 10 (Nov. 2006), 1190–1199.
11. Knuth, D.E. and Daylight, E.G. *The Essential Knuth, volume 2013.* Lonely Scholar, 2013.
12. Petzold, C. *The Annotated Turing: A Guided Tour through Alan Turing's Historic Paper on Computability and the Turing Machine.* Wiley Publishing, Inc., 2008.
13. Priestley, M. *A Science of Operations: Machines, Logic and the Invention of Programming.* Springer, 2011.
14. van Rijsbergen, C.J. Turing and the origins of digital computers. In *Aslib Proceedings 37*, pp. 281–285. Emerald Backfiles, June/July 1985. Paper presented at an Aslib Evening Meeting, Aslib, Information House, March 27, 1985.
15. Vardi, M.Y. Who begat computing? *Commun. ACM 56*, 1 (Jan. 2013), 5.

**Edgar G. Daylight** (egdaylight@dijkstrascry.com)—also known as Karel Van Oudheusden—has a Ph.D. from KULeuven in Belgium and is a researcher in software engineering and the history of computing at Utrecht University in the Netherlands.

The author encourages readers to view his blog post at http://www.dijkstrascry.com/ATuringTale for more detailed information pertaining to the references cited in this Viewpoint.