Grace Hopper's Contributions to Computer Science

Andreas Moritz

May 26, 2015

1 Introduction

Grace Murray Hopper lived from 1906 to 1992. She earned her PhD in mathematics in 1934 at Yale University. Hopper received numerous awards and honours throughout her lifetime such as:

- The first annual 'Computer Sciences Man-of-the-Year Award' from DPMA
- Fellow of the IEEE
- Fellow of the AAS

But what did she accomplish to receive those honours? This document gives an overview of her most influential and remarkable contributions to computer science.

2 First Contact with Computers—The Harvard MARK Series

In 1943 Grace Hopper joined the US Naval Reserve during World War II.

She was assigned to code the newly built MARK I (also known as the IBM Automatic Sequence Controlled Calculator) at Harvard University under the lead of Howard Aiken. Hopper was "the third programmer on the first large-scale digital computer"[5] as she liked to being introduced as when giving speeches.

The MARK I computer used a punched paper tape for instructions, where holes would represent a '1' and paper a '0'. Hopper's first computing publication was a 500p. coding manual for the MARK I called "A Manual of Operation for the Automatic Sequence Controlled Calculator". She continued to work on the MARK II and MARK III under Howard Aiken.

3 The First Compiler

In 1949 Grace Hopper "decided to join the Eckert-Mauchly Computer Corporation to help design and program UNIVAC I, the first commercial electronic computer."[3]

In 1951 she was given the assignment to put all pieces of coding together that they had developed over these two years, so that a library of sub-routines was available that everyone could use. Reusing a sub-routine in a new program involved copying it by hand and adding an offset to the addresses to position it correctly in the new program. However, Hopper found out that "Programmers couldn't copy things correctly"[3] and "Programmers couldn't add."[3] Hence she made this process automated on the UNIVAC "whose sole job was to copy things and do addition."[3] This was the birth of the first compiler, known as A-0 (Arithmethic Language 0).

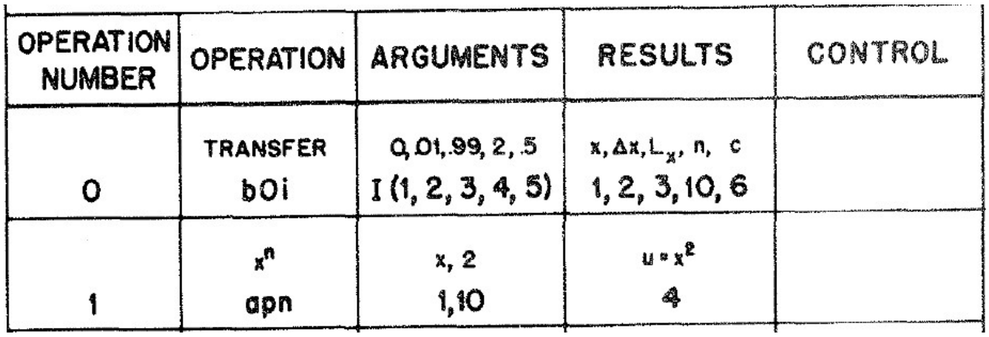
"A major reduction in time consumed and in sources of error has been made. If the library is wellstocked, programming has been reduced to a matter of hours, rather than weeks. The program is no longer subject either to errors of transcription or of untested routines."[2]

Although the A-0 compiler was more of a linker than a compiler concerning today's standards, it was quite revolutionary in its time. It enabled the programmer to call a sub-routine from a library by using call-numbers in the program.

The call-numbers consisted of alphabetic characters, which determined the class of operation (e.g. a = arithmetic, x = exponential, amc = arithmetic multiplied by constant,...). The symbols for the arguments and results were stored in a numbered list and would be referenced by their corresponding number. The alphabetic characters of the call-number were followed by the arguments that the sub-routine should take and the result-numbers in which the sub-routine should store the result. This allowed every programmer/mathematician to use their own symbols. [2]

In Figure 1 an example is shown how to calculate x^2. It is part of a larger example shown in "The Education of a Computer" [2, figure 9].

Figure 1: A-0 Compiler example, store x^2 in u — x has number 1 and u has number 4 in the symbol list, which is not shown here



OPERATION NUMBER	OPERATION	ARGUMENTS	RESULTS	CONTROL
0	TRANSFER bOi	Q,01,.99, 2, .5 I (1, 2, 3, 4, 5)	$x, \Delta x, L_x, n, c$ 1, 2, 3, 10, 6	
1	x^n apn	x, 2 1,10	$u = x^2$ 4	

4 Rise of the High-Level Programming Languages

Hopper's group would make numerous improvements on the A-0 compiler. This led to the A-2 compiler in 1953. "It used short mnemonics for operation; for example, the statement ADD 00X 00Y 00Z would add X and Y and put the result in Z."[1] The A-2 Compiler was the first Compiler that was used extensively by various UNIVAC users and paved the way for various programming languages.

In 1954 Grace Hopper was named Director of Automatic Programming for the UNIVAC division. The idea of more human-like programming languages began to catch on, main benefit being lower cost and higher speed of developing computer programs. Various programming languages were under development at the same time, intended for different communities. For scientific applications, FORTRAN by IBM and AT-3 (MATH-MATIC) by Hopper's Group. For computer scientists, ALGOL by a committee of European and American computer scientists. For AI researchers, LISP at the MIT.

Hopper also had a group of non-scientific people in mind though. Business data processing customers. She "knew darn well most people didn't like symbols. And you had to give them something else to work with, and obviously the next-best thing was English." [3]

Thriving towards an English programming language, Hopper and her staff developed B-0 (Business Language 0) in 1955, also known as FLOW-MATIC. It allowed the description of user-defined data types by filling in pre-printed forms. The operations (e.g. DIVIDE, SUBTRACT, MOVE) as well as the data names (e.g. UNIT-PRICE, DISCOUNTS, INVENTORY) were written in English syntax. B-0 was the first programming language to separate different parts of the program into different sections. The definition of the data types was done in the Directory section while the actual execution statements were written in the Compiler section.[6]

Hopper got rid of the various operation symbols from the other programming languages by replacing them with imperative verbs. For example 'B = A' would become 'TRANSFER A TO B'.

FLOW-MATIC's English syntax was quite groundbreaking because people commonly believed that computers could not understand English. This was the main reason for its great success and influence on COBOL.

5 Common Business Language COBOL

In 1959 Hopper suggested with some colleagues a common business language. The main purpose was greater compatibility between different systems. This would save a lot of time and money, because programs would not have to be completely rewritten upon upgrading the hardware. She convinced the Department of Defense to organize the Conference on Data Systems Languages (CODASYL) to consider the specifications of a common business language. Grace Hopper served as technical advisor of CODASYL and as such did not actively develop COBOL, but rather gave general guidance.[4]

This resulted in COBOL being strongly influenced by her ideas and previous work on

FLOW-MATIC (e.g. the English syntax and the separation of the code into different sections). Although the Commercial Translator (COMTRAN) from IBM also had influence on the development of COBOL, it was comparably small. According to Hopper it only had any influence at all to avoid the risk of IBM continuing their development on COMTRAN, which would have been a direct contender to COBOL.[3, p. 37]

The first specifications of COBOL were presented in 1960 as 'COBOL-60' and various computer companies began to provide compilers for it. Over the years COBOL was updated many times. This resulted in incompatibility between the different versions. Grace Hopper expressed the need for standardization and was a huge influential factor for the first standardization of COBOL in 1968.

The great success of COBOL was mainly due to its high portability, standardization and influence by FLOW-MATIC. One of the initial developers of COBOL, Jean Sammet, said "Without that existing practical use of FLOW-MATIC I doubt that we would have had the courage to develop a language such as COBOL."[4]

6 Conclusion

Grace Hopper was one of the most influential women in computer science. She was greatly involved in the development and coding of the first digital computers and produced the first compiler. She developed and promoted the first English programming language for business data processing. This was probably also her most influential work due to the role it played in the development of the first highly machine-independent programming language.

References

[1] George T. Gray and Ronald Q. Smith. Sperry rand's first generation computers, 1955-1960: Hardware and software. *IEEE Ann. Hist. Comput.*, 26(4):20–34, October 2004.

[2] Grace Murray Hopper. The education of a computer. In *Proceedings of the 1952 ACM National Meeting (Pittsburgh)*, ACM '52, pages 243–249, New York, NY, USA, 1952. ACM.

[3] Angeline Pantages. Oral history of captain grace hopper. pages 1–54, 1980.

[4] Jean Sammet. Farewell to grace hopper – end of an era! *Commun. ACM*, 35(4):128–131, April 1992.

[5] Jean E. Sammet. History of programming languages i. chapter Introduction of Captain Grace Murray Hopper, pages 5–7. ACM, New York, NY, USA, 1981.

[6] Wikipedia. Flow-matic. [Permalink; accessed 25-May-2015] `http://en.wikipedia.org/w/index.php?title=FLOW-MATIC&oldid=659142757`.