Lisp machines and the analysis of their high-level language computer architecture

Dominic Dabish and Evan Bradley
School of Engineering and Computer Science
Oakland University, 2200 N Squirrel Rd, Rochester, MI 48309
{dadabish, edbradley}@oakland.edu

***Abstract*—The high-level capabilities that Lisp affords programmers makes it very useful in certain applications, but also causes it to run slow on most computers, which were designed for different types of executions. To account for the incompatibility between Lisp and the hardware on which it is executed, computers called Lisp machines have been designed specifically to run Lisp programs by designing the processor hardware to accommodate the language features Lisp possesses. Achieving greater efficiency in running Lisp was accomplished using various methods, both in the hardware and in how data would be represented in the software. Hardware alterations largely consist of changes to the circuitry in the processor itself as well as the number and purpose of processors in the machine. Software changes mostly correspond to the representation of lists, the main data structure in Lisp. Lisp machines were largely deprecated in the 1990s, giving way to traditional computers which were fast enough to run Lisp with reasonable speed. Despite this, Lisp machines remain a popular subject for study and discussion amongst those who are familiar with Lisp.**

Keywords*—*Lisp, Lisp Machine, computer engineering, vintage computing, computer organization.

I. Introduction

The history of Lisp begins with Alonzo Church's introduction of lambda calculus in the 1930s, where Church attempted to reconcile mathematics and computation. Lisp takes many notational cues from lambda calculus, and in a sense serves as a practical implementation of the ideas Church presented. Lisp machines were created as a response to the decreasing costs and increasing complexity of computers in the 1970s, and were rendered virtually defunct in the 1990s by the same conditions that had created them.

A. Early history of Lisp

John McCarthy created Lisp while working to create a new computer language to replace languages like the Information Processing Language (IPL) and Fortran, which he saw as cumbersome and insufficient for programming artificial intelligence (AI) [1]. He began work on what was then called lisp LISP, or the LISt Processing language, in 1958 while working at the Massachusetts Institute of Technology (MIT), and published a Communications of ACM paper in April of 1960 that detailed the basic workings of the language he constructed [1,2]. By April 1958, Steve Russell had implemented some Lisp functionality in machine language on the IBM 704, much to the surprise of McCarthy, who believed his work to be entirely theoretical before seeing the implementation. Development of Lisp began to occur in a more concrete fashion after the actual implementation of Lisp code on the IBM 704. McCarthy and his team then began to focus on creating a usable version of Lisp in addition to creating something that works in theory [1].

In the late 1960s, Jon L. White published a paper describing MACLISP, a variation of Lisp that ran on the Digital Equipment Corporation's PDP-6 [1]. This was followed by the creation numerous other Lisp dialects by individuals and groups who were not directly related to McCarthy. Many of these were combined into Common Lisp in the 1980s [Wikipedia]. One of these dialects was Scheme, which, along with Common Lisp, continues to be one of the two major Lisp dialects to date. It was created in 1975 by Guy L. Steele and Gerald Jay Sussman at the MIT AI Lab, and on whole is much simpler than Common Lisp.

B. History of Lisp machines

Lisp machines were some of the first single-user workstations, which at the time were an exception to the common timesharing systems where multiple users would connect to a single machine through terminal stations. This change came in part due to Lisp's nature of consuming the entirety of the computer's resources, which made its use difficult on multiple-user systems [3]. Two main companies emerged from the increasing demand for Lisp Machines: Symbolics and Lisp Machines, Inc. Larger companies began to produce Lisp Machines as well, including Xerox and Texas Instruments.

These machines were often seen as valuable for not just their ability to run Lisp, but for the advanced features they offered in terms of graphical user interfaces (GUIs) and programmability due to the flexibility that Lisp afforded programmers. Despite their advances in usability, they were often behind in terms of hardware due to their niche audience, they sold to a much smaller market than general-purpose machines, which quickly raced past Lisp machines in terms of hardware capability [4]. Eventually general-purpose computers became faster at

This paper was submitted for review on March 25, 2016. It was produced with help from Oakland University's subscription to publication sources with information on Lisp and Lisp machines.

Dominic and Evan are Computer Science majors at Oakland University in Rochester, MI, 48309.

running Lisp despite the hardware incompatibilities, and in combination with the slow development of AI in the early 1990s, most Lisp Machine vendors either left the market or went bankrupt.

II. The Basics of Lisp

Appendix A
Proof of the First Zonklar Equation

Some text for the appendix.

Acknowledgment

The authors would like to thank...

References

[1] H. Kopka and P. W. Daly, *A Guide to LaTeX*, 3rd ed. Harlow, England: Addison-Wesley, 1999.