Humboldt-Universität zu Berlin
Mathematisch-Naturwissenschaftliche Fakultät
Institut für Informatik

Drift: an imperative programming environment for the cloud

Masterarbeit

zur Erlangung des akademischen Grades
Master of Science (M. Sc.)

eingereicht von: Frank Lange
geboren am: 08.08.1989
geboren in: Berlin

Gutachter/innen: Prof. Dr. Jens-Peter Redlich
Prof. Dr. Joachim Fischer

eingereicht am: ..... verteidigt am: .....

Contents

1 Introduction **6**
1.1 Goal of this work 6
1.2 Structure of this work 6

2 Distributed Programming **6**

"If I had more time, I would have written a shorter letter."

– Blaise Pascal

1 Introduction

...

1.1 Goal of this work

...

1.2 Structure of this work

...

2 Distributed Programming

In 1937 a paper titled *"On computable numbers, with an application to the Entscheidungsproblem"*, written by Alan Turing, was published [1]. In it Turing defines what is now known as a *Turing machine*, a theoretical machine model for executing arbitrary computation. He demonstrates how to formulate instructions for this machine in order to configure its execution behavior and goes so far as to show how this machine could even receive another machine configuration as its input and then execute what the other machine would have done, thereby effectively emulating the given machine.

But he wasn't the only one thinking about computation. As summarized by Denning in [2], Kurt Gödel, Alonzo Church and Emil Post all contributed to the task of exploring the boundaries of computation, setting the tone for the next 80 years of research concerning *computation*, *computers* and *automation*.
Of all the proposed approaches of how to tackle the concept of computation and turning it into a tool that could be understood and used by delivering a theoretical framework, one could argue that Turing's machine model emerged as a winner because, although not at the time, it turned out to be closest to what machines became to be. How they were structed and how they basically operated. Still, to this day.
So in order to use or *program* these machines, these computers, programming languages were developed and over the decades have gone through their own evolutionary process [3] with the first generation using binary machine instructions to todays languages that allow for higher level programming styles like *object oriented programming* [4] or *functional programming* [5].

But no matter what mainstream programming language one looks at today, they all seem to have one thing in common, which also gets replicated by each new language that arrives: they are all focussing on describing computation. Computation executed by a single core machine. Because for most of the last century, this was the problem we were facing.

However, with the advent of commodity multicore hardware [6] the problem domain changed. Now a machine isn't a single machine anymore, but rather a group of multiple CPU cores that can operate and compute independent of each other.
Unfortunately mainstream languages are still struggling with delivering language concepts and semantics that allow for utilizing this new hardware. In Fig.1 one can see a very minimal multithreaded program written in Python, one of the most used languages today [7].

```
from threading import Thread

def count(n):
  while n > 0:
    n -= 1

t1 = Thread(target=count,args=(1000000,))
t1.start()

t2 = Thread(target=count,args=(1000000,))
t2.start()

t1.join()
t2.join()
```

Figure 1: Python multithreading example

As was shown by Beazley in [8] this multithreaded version is up to *two times* slower than the single threaded version and the reason for this is the so called *Global Interpreter Lock* (GIL) used by the Python interpreter which basically prevents multiple native threads from truly executing in parallel.
This shows how important it is to reevaluate exisiting programming languages and methodologies whenever the underlying machine or system changes dramatically.

Another such paradigm shift was the advent of the *internet*, or large scale networking in general, because it dawned what is now known as the *information age*. Although this is a rather broad term, describing global change in almost all areas of modern civilization, I believe these changes can also be seen in the area of computing and computation. Because in order to deliver products like: the world wide web, internet search, internet advertising, social networks, social media, music streaming, video streaming, messenger services, navigation/maps, e-commerce etc. companies had to build massive computer networks in order to either store massive amounts of data or to simply scale their product to millions or billions of users. And altough almost all of the current products, apps, or services could not exist, without the computation that is implemented inside their core parts, it is *communication* that has become the main obstacle for building large scale *information systems*.

References

[1] A. M. Turing, "On computable numbers, with an application to the entscheidungsproblem," *Proceedings of the London mathematical society*, vol. 2, no. 1, pp. 230–265, 1937.

[2] P. J. Denning, "What is computation," *Ubiquity*, 2010.

[3] Wikipedia, "Programming language generations — Wikipedia, the free encyclopedia." `http://en.wikipedia.org/w/index.php?title=Programming%20language%20generations&oldid=753335536`, 2017. [Online; accessed 02-February-2017].

[4] B. Stroustrup, "What is object-oriented programming?," *IEEE Softw.*, vol. 5, pp. 10–20, May 1988.

[5] P. Wadler, "The essence of functional programming," in *Proceedings of the 19th ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages*, POPL '92, (New York, NY, USA), pp. 1–14, ACM, 1992.

[6] S. Olsen, "DailyTech - Here Comes Conroe." `http://www.dailytech.com/HereComesConroe/article3228.htm`, 2006. [Online; accessed 02-February-2017].

[7] TIOBE, "TIOBE Programming Language Index for January 2017." `http://www.tiobe.com/tiobe-index/`, 2017. [Online; accessed 02-February-2017].

[8] D. Beazley, "Understanding the python gil," in *PyCON Python Conference. Atlanta, Georgia*, 2010.

Selbständigkeitserklärung

Ich erkläre hiermit, dass ich die vorliegende Arbeit selbständig verfasst und noch nicht für andere Prüfungen eingereicht habe. Sämtliche Quellen einschließlich Internetquellen, die unverändert oder abgewandelt wiedergegeben werden, insbesondere Quellen für Texte, Grafiken, Tabellen und Bilder, sind als solche kenntlich gemacht. Mir ist bekannt, dass bei Verstößen gegen diese Grundsätze ein Verfahren wegen Täuschungsversuchs bzw. Täuschung eingeleitet wird.

Berlin, den February 2, 2017 .....