Robert Alm
Movant Lund
Lund 2017-07-21
Extended Essay

Title: My participation on Google HashCode 2017

Resume:This documentation is about my participation on the competition Google Hashcode 2017, my preparation for the Google HashCode 2017[1] and Google CodeJam 2017[2], and my efforts to create prefabricated code for the two contests and the programming contest of Code Chef[3]. This documentation is a project in order to complete my education as Network Technician on Movant private technical school[4]. I will try to describe the contests, the issues with the competitional programming and the programming contests and the preparation and actual participation on Google Hashcode 2017

I had the honor to take part in two teams. The team ”Evil Chickens of Doom” which had to cancel its participation for technical reasons and after that to the team ”Team FF” that unfortunately failed on the online qualification round, but delivered results on the online extended round.

THE COMPETITIONS

Google Hash Code

Google Hash Code is a developing competition that takes part in Europe and invites participants only from Europe, Middle Asia, and Africa. In every team is mandatory to be 2 to 4 participants and the important aspect of the competition is that the participants have to solve real problems world problems, more specific problems that software engineers are facing in big companies like Google.

Google Cope Jam

Google CodeJam is probably the most known programming contest of google and maybe the most known programing contest in general. Google CodeJam accepts contestants from any background and it is famous for the relatively easy qualification rounds. Thousands of participants are taking place every year, and it remains an active subject for discussions for the whole year.

Code Chef

Code Chef is an initiative that promotes the idea of learning programming on India and it is turned to be one of the most active programming competitions globally. It has an complete system for programming training, a very active programming forum, that can compare sometimes to Stack Overflow, (Stack Overflow if the most active programming forum globally), with a community that no other programming competition was able to achieve. It organizes monthly competitions, sometimes few annual competitions as well, and allows smaller pioneers to promote their competitions through Code Chef.

THE PREPARATION FOR GOOGLE HASH CODE

Disclaimer: Before I continue I should point out that when I speak about preparations I mean the preparations that are made by the first team with the name “Evil Chickens of Doom” and mostly those which I participated. All the members of both teams were made preparations that it was not possible to collected and be included on my documentation

My participation to the Development Competition Google HashCode was in my resolution list for 2017 and I was determined to go as well as possible. So it was clear that I had to be prepared as much as possible in order to achieve the desired results. Actually a programming competition and even worse a development competition, depends from the preparation of every contestant.
My preparation was about choosing a language and IDE, forming a team, and train alone and with the team.

The language and the IDE

The first big issue of my preparation was to choose a language and to find an suitable IDE, and to learn language and the environment as much as possible. My choice for a language was Python. Python is useful as an language because of its dynamic character. To be honest Python and R[5] was my two choices, Python for a more programming oriented approach or R for a more Query oriented approach. I was more fun of the programming approach, (I am not big fun of query languages), so I choose Python. I chose Python 3.5[6] because it was the newest stable version of Python, despite the fact that I was feeling more comfortable with python 2.6

For IDE I did the mistake to choose Aptana studio 3.0[7]. I choose Aptana because it is based on Eclipse, an IDE that I respect, because I heard good things about it, because it was open source and because I had access to well made tutorials on Aptana 3.0 by Lynda.com[8]. My choice was wrong and it took me a fair amount of time to admit it. The people that were responsible for the administration of Aptana repository on Github[9] were unresponsive and the community of Aptana were not able to add patches and fixes/ The result of the absence of the administrators was an Aptana that was outdated broken,a and disconnected from its modules. Still a bad ass IDE, but unfortunatelly broken.

My preparation on the matter of the language was to refresh my knowledge on Python by using tutorials from Lynda.com, (through Linkedin) and by making practice on Code Chef.
The tutorials that I used to get in touch with Python were:
1)Learning Python by Joe Marini[10]
2)Programming Foundations: Real-World Examples by Barron Stone[11]
3)Python 3 Essential Training by Bill Weinman[12]

Eventually, at the last moment, I changed my IDE to JetBrains' PyCharm[13], a very effective and

functional IDE that unfortunately I was not educated enough to use efficiently. (but I will return to that later).

Forming a team

One of the most difficult tasks that participants without team are facing on Google HashCode is to find a team. The organizers of the competition give a great effort, (with threads on social networks and forums), to help the participants to find a team but that doesn't mean that finding a team is that easy. In order to find a team someone has to choose a programming language that is used by many contestants, to find the right places to search, to search actively and most at all to avoid beginners on programming at any cost.

By avoiding beginners I don't mean that on some kind of elitism and I really believe that it should be given a chance for programmers with less experience by those who already have collected some experience, (with some kind of mentoring and help), but the beginners and the “jobseekers” are becoming a plague for competitions especially for those who are made by Google or Facebook. Big companies like Google and Facebook are giving an another prestige to the competitions and the problem is that there are people that are wish more to get hired by google or to be called programmers, (or successful founders or CEO), instead of the actual pleasure and challenge of writing a code.

Another issue was that we had to use the exactly the same language and the same version in order and to use exactly the same tools of cooperation, (I mean Git[14] and GitHub mostly) in order not to spend more time to choose tools and to learn how to use them. Something that I found out later was that it was important as well to have the same IDE and the same version and packages. To have the same environment makes sense now but back then I failed to figure this out.

In order to solve all those issues, in my announcement I requested for people that were fluent with Python 3.5 with a GitHub account and a good knowledge of GitHub, (I insisted on GitHub because I know that the installation of a adhog git server can get very messy very fast and the possibility to get stuck on some technical issue in the middle of the competition was not so pleasant thought). A last thing was that the participants in the group had to be able to communicate in English for obvious reasons.

By the fist wave of announcements we were able to form a team with 3 members. We called ourselves “Evil Chickens of Doom” we installed out communication/cooperation tools and we continue to searching for the fourth member of our team. The one of the guys, (I don't say the names of the participants for reasons of respect to their privacy), was a fan of Python 3.5 and he was spending his free time on participating on programming contests. The other guy was experienced open source programmer who enjoyed mostly to build features for Linux by using Python. For various reasons the team didn't work, (for personal issues mostly, we didn't had any fight the time that we work together), and it was dissolved few days before the competition.

A day before the I got contacted from a guy from Switzerland who as it seems was way more experienced programmer that I, and he bring me in contact with one other experienced programmer from Austria. If I recall correctly both are professors on data science and they are heavily involved on computational programming. From my side I tried to contact the one of my old team that he may was available but he wasn't. We called ourselves TeamFF.

Practicing coding

The online qualification round for Google HashCode 2017 was planned for the 23rd of February while the whole preparation progress started, (for me at least), on the 1st of January. While I was determined to study the language, (Python 3,5), it was seemingly the best solution for everyone to prepare alone until the 1 of February. In that way every participant had a fair chance to refresh his knowledge on Python and to prepare himself as he saw fit.

The plan was for everyone to practice competitive programming by himself until the first of February and after that we would be more able to do some training together, (team training between individuals that don't know each other could be difficult in some cases so it had to be only in our last phase). Unfortunately we didn't had the chance to to do team practice.

My practice in coding for both of the two months was solo and I took the decision to train in Code Chef instead of trying to solve last problems on Google HashCode or Google CodeJam pages. The reasons was that problems in Code Chef where better organized, in a better platform, and of course behind Code Chef is an extremely active community that analyzes every aspect of competitive programming. Also the available last problems from Google HashCode were just six and they meant to be for teams of 2-4 persons. If I tried to solve any of those problems alone, that could take me lot of time and energy, that I was planning to use more effectively, and I was waiting that we will practicing those problems together as a team after the 1st of February, something that unfortunately it wasn't the case.

The main difference between the google competition platform and Code Chef's platform is that Google uses a simpler, (and somehow more effective), submission system in which the participant requests for the input values in a text form, (usually an input.in file), and he has to upload the text file with the output values in a certain amount of time.

The submission system on Code Chef is fancy and complicated, (and bit buggy sometimes, unfortunately), in which the participant uploads the code instead of the solution and the platforms runs the code. As I said sometimes it gets bit buggy, but I can imagine the reason why the choose to use that kind of submission system. The wanted to offer to their users an easy programming environment away from unnecessary compilers.

Analysis of the old problems

In order to be as prepared as possible for the online qualification round of Google HashCode 2017 we decided to check out the older problems from the previous competitions. We worked on two different issues. The main template with the more trivial issues (like I/O and data structures) and the heart of the problem with the mechanism that can solve the Google HashCode problems.

The template was about the trivial stuff. Even if they are trivial, the issues that are included when someone designs the template are important to be solved, and it is a fact that the majority of the contestants on programming competitions have lost at least one competition because of the trivial issues. Even if someone is so skilled that he can implement the template without bugs, the time is an important factor, (the energy as well), in the middle of the competition and it shouldn't be wasted on creating the template or checking the details.

Google the last years has the habit to make the I/O bit more difficult by divining the input stream to cases, which becoming with the time even more complex. For someone who works with queries, (data analysis methods), it is probably easier to handle the input stream but when someone works

with the default version of a language then things can get bit hairy. I was not able to figure out to which module I can use to handle the issue with the stream, so I decided to build my own prefabricated code for this issue which I called “caser function” and it was the main feature of the prefabricated template, (optimal template).

The heart of the problem was not so mysterious to as but we had to be as prepared as possible to face any possible problem, We failed miserably on that but our thinking was right. We analyzed the previous problems and we figure out that it would be better to use object oriented principles in order to implement some kind of independent objects that could solve the problem from their own perspective. At this point I should point out that in the way that we handled the issue we were very inspired by the “Design patterns”[15] book, ("Design Patterns: Elements of Reusable Object-Oriented Software", published in 1994 by the so-called "Gang of Four"), and we tried to implement our own Pattern.

Our design pattern was made by eggs chickens and the cote. The chickens were the moving objects, function objects that were able to move independently and take decisions accordingly to the input data that they are receiving. The egg would be the objects that would be generated by the chicken objects and probably function as nodes. The cote objects was meant to be used as nodes or as a playground for the chicken and egg objects. At last an idea that we had was to make the most complicated prefabricated functions as function classes. But we didn't had the time so we had to let it be.

Optimal Templates

One thing that I really invested a lot of time and effort was the idea that most of the elements on a programming competition's template can be prefabricated. Since I started practicing on my coding skills, I was working same time on the creation of an optimal template.
The basic idea is that we can have ready the small issues, like the typical input/output in order to avoid to waste time on those and it is a good opportunity to work on the quality of the code, improve structure and the syntax, make the variables more relative and add some helpful comments. From my whole effort I managed to have to have three different versions of what I call “Optimal Template”.

One template is for the Code Chef and it is the most tested right now, and as I see it works fine, as it saves me really a bunch of time when I am practicing on problems of Code Chef. One other version that I have is for Google CodeJam and it works as well on an optimal way.
For Google HashCode, even if I did the most work on that competition I am not sure yet what it can be considered optimal. I am sure it is a good base to work on, but it is not ready. It will take a lot of time and energy in order to end up with something presentable.

What went wrong on the preparation

There are many the mistakes that took place on the preparation, for example when we weren't figured out the necessity of using the same environment and same IDE, but it would be easier and more beneficial for the reader if I focus on what was that did really wrong on our preparation.
The biggest mistake at all was that we didn't invest time to work more on queries. It wasn't exactly mistake but more was some kind of incapability as we were educated for a more default programming style and not queries.

At this point I believe that it is a good time to leave my rant about Google Programming

Competitions and the majority of programming competitions worldwide.
Programming competitions all over the world are starting the last years to following a very specific motive. It is usually an input in a form of stream that gives some kind of values separated in cases and it is expected that the participant will return as output the result of different mathematical or logical decisions.

That is great for those that are working on computational programming, machine learning, and data analysis but the whole world is not just queries and formulas that are based on discrete optimization. Of course I understand that Google HashCode is based on real life problems and of course I agree with the idea that the majority of the issues that Computer Science faces is about data queries and discrete optimization.

The thing that I am trying to point out is that even in data science that everything is about a good query, even there the very basic principles of the Data science are the three V, (Volume, Velocity, and Variety), and as the programming cynically turns from advanced spaghetti lines to be advanced spaghetti queries we tend to forget that data is not only about the volume and the velocity is about the variety as well.

Organizations that raise awareness on programming are usually claiming that programming is not about tons of theory, but it is about skills and creativity and the competitions are claiming that are for everyone, (and they are really are), and everyone regardless his background should come to check his skills.

On praxis we see two kind of people in programming competitions. Those who are using their knowledge and their inside knowledge and they just trying calmly different versions of queries with a coffee in their hand in order to achieve better velocity on their result. On the other hand we have the newcomers that are getting demolished, as they are trying with traditional programming to survive in the happy place of the fast results. That is like taking a knife to a gunfight.

Probably I am wrong on that and I am seeing the thing in a negative way, I just like it more when programming looks like programming instead of data analysis, especially when we are talking about competitions. I really understand that data analysis is the new hot thing this period, just I wanted to point out that programming is about building things and not just optimizing data queries.

THE SHORT PREPARATION BEFORE THE COMPETITION

As it is known the last first team was dissolved and replaced by the second team with the name “TeamFF”. After we got to know with each other and we decided that we will try to work together, we started to do the necessary preparations.

The first decision that we took was to install a virtual environment with Anaconda for python 3.5[16]. Anaconda is an environment for python with modules that are designed to work for work that scientists are usually do (data handling). In that point I discovered that I had to change my IDE asap. Aptana studio 3.0 wasn't in position to handle many paths and many modules so I had to use PyCharm instead of Aptana.

The most experienced person in our tema created a list of which modules we had to install. The modules were many but the one that we really used was numpy[17]. Numpy is an important package for scientific computing with Python. Unfortunatelly it was something that I didn't new so I had to do some quick learning on Numpy. For someone with experience on SQL it gets bit more

easy to get the hang of it, but even so. It was clear for that I wouldn't be able to produce much result with it, so I had to step bit back.

The decision that I took, was to step back and to have ready the prefabricated template in order to manage the trivial issues and help the others to focus on the problem.

The online Qualification round – 23/02/2017

We lost. A failure on my caser because of a bug and the long time that it took me to recover the damage played an important role to our failure.

The long qualification round

The philosophy of the long qualification round is to give the opportunity to those that were not able to solve the problem to solve it and to give them as well the opportunity to compare with each other just for fun, in a fairly longer time period. We decided to try it anyway. The other two of the team used the time to try and build better and faster algorithms while I decided to face my demons, to fix the flaws of my template, and to experiment a bit with the code that I had on my hands.
We delivered and our rank was somewhere in the middle of rankings.

CONCLUSIONS

A failure has always a bitter taste and it is a sad thing when someone spends time and effort to document his failures, but it was kind of important as as spend 2 months of hard work on that project and even if it was a failure. There are so many variables that I had to analyze somehow what I did.

My rant about the spaghetti queries left on the table so I believe that I don't have so many conclusions left. This period of my life I am trying to study bit more about data science. I don't like that much data science but I have to admit that I am impressed from the art of data handling and I just had to learn more about it. That means that I will have to learn and practice numpy and I may try again to run for the Google HashCode 2018 with the same persons as the last time.

I can understand that the theory that I need to study right now is discrete optimization[18] and machine learning[19], (apart of my efforts to study the art of data handling). An I need to go deeper, and start to work bit more on my code. There are many prefabricated functions that they could be better on a generator class and I am not feel as expert on design patterns so I will have to be bit more around object oriented programming.

A very last issue is my the form my syntax. I noticed that I am using the programming style that I use on Javascript, so I will have to invest bit time on PEP8[20] programming style for Python.

LINKS

[0]the repository with the project
https://github.com/Exarchias/Google-HashCode/tree/master/Google%20Hash%20code%202017

[1]Google HashCode 2017 homepage
https://hashcode.withgoogle.com

[2]Google CodeJam 2017 homepage
https://code.google.com/codejam/

[3]Code Chef homepage
https://www.codechef.com

[4]Movant private technical school homepage
http://movant.se/vara-skolor/lund/

[5]R - wikipedia
https://en.wikipedia.org/wiki/R_(programming_language)

[6]Python 3.5 homepage
https://www.python.org/downloads/release/python-350/

[7]Aptana studio 3.0 homepage
http://www.aptana.com/products/studio3/download.html

[8]Lynda.com homepage
https://www.lynda.com

[9]Github homepage
https://github.com

[10]Learning Python by Joe Marini
https://www.linkedin.com/learning/learning-python

[11]Programming Foundations: Real-World Examples by Barron Stone
https://www.linkedin.com/learning/programming-foundations-real-world-examples

[12]Python 3 Essential Training by Bill Weinman
https://www.linkedin.com/learning/python-3-essential-training

[13]JetBrains' PyCharm homepage
https://www.jetbrains.com/pycharm/

[14]Git homepage
https://git-scm.com

[15]Design Patterns - Wikipedia
https://en.wikipedia.org/wiki/Design_Patterns

[16]Anaconda for python homepage
https://www.continuum.io

[17]Numpy homepage
http://www.numpy.org

[18]discrete optimization - wikipedia
https://en.wikipedia.org/wiki/Discrete_optimization

[19]machine learning - wikipedia
https://en.wikipedia.org/wiki/Machine_learning

[20]PEP8 homepage
https://www.python.org/dev/peps/pep-0008/