CS 6704: CTTC 4

Sourabh Shetty

Exploring Regular Expression Usage and Context in Python

The authors here chose to solely focus on projects that were written in Python. While I normally tend to critique the limited scope of a research's language selection, here I think focusing on just one language actually improved the quality of the research rather than hamper it. For a topic like regular expressions, looking at a breadth of languages would've unnecessarily complicated the data to be analyzed.

I didn't like that the research survey was only conducted on 18 people from the same small company. In companies people often end up adopting similar coding practices and conventions. When I used to work I noticed people sometimes even abandoning styles they were used to, to conform to previous code examples written by other, usually more senior employees. At one point I noticed a newer employee reusing my own flawed regex expression for a phone number that I myself had written months previously and had forgotten to correct. People working at the same company, especially a smaller one, will often end up having similar experiences and will thus not be a true representation of how people at a more general level actually use regular expressions. I think the authors should have surveyed on people across a variety of companies as well as independent developers. Since they already had a mined data from projects, they could have sent out the survey to the developers who committed the regular expressions that they had mined. Even if 0.1% of the people had responded it would have been a better sampling of people. The authors have included an acknowledgement of this, and say that the developers' past experiences makes up for the fact that it was conducted at a small company, but I disagree with this, and I think that they could have a put a little more effort into finding developers to take the survey.

Aside from that I did think that the work put into the data aspect of the research was great and had tremendous merit.

Exploring Regular Expression Comprehension

I found this to be one of the most exciting papers I had read so far in this course, and found it to be quite important and meaningful. I have observed code smells strongly impacting readability to a point where I knew people who would avoid regular expressions at all costs due to the fact that their limited interaction with regular expressions had been with hard-to-understand expressions. Only when regular expressions had been explained to them from scratch did they begin to understand them and realize their power.

I also normally tend to not understand a lot of the math in research papers, but here the math was not just understandable, but also was interesting to read through.

I found the authors approach to ranking the ideal understandability of regular expressions to be very inspired, and I enjoyed reading the 'Results' section, since it very clearly depicted both, the ideal understandability ranking of the given expressions, as well as what actually was more popularly used in practice, and gave a plausible reasoning as to why that was the case.

I do agree with the authors claim that this research can indeed guide the design of regular expressions.

How well are regular expressions tested in the wild?

This paper was a very fascinating read for me. While I consider my debugging skills to be pretty good, I admit I am very lazy when it comes to testing. I personally have very rarely tested my regular expressions, but I had always assumed that was a function and a natural extension of my existing laziness. Reading this paper, however, made me realize how widespread the issue was and that even seasoned developers often neglected testing their regular expressions.

One issue I had with the paper was that sometimes, regular expressions may be used for something very small for which having an additional test case doesn't make sense since there isn't as much ambiguity. There might not be any avenues for it to fail if it is simple enough, so a developer may have consciously decided not to have separate test cases for them. The lack of testing as described in the research implies possible problems, but in my opinion when the regular expression is simple enough, testing might not be required in the first place. The mining process just looked for wherever the regex functions were used rather than the complexity of the expressions themselves.

I think the research could be further improved by including the complexity of the expressions as a factor to be considered. A more complex regex pattern would require greater testing whereas a simple pattern like 0* or [0-9] wouldn't require as much testing. Thus I believe accounting for this is important.