CS 6704: CTTC 11/12

Sourabh Shetty

Repairing Programs with Semantic Code Search

I found this to be an excellent topic of research and matches a lot of my interests. One of my projects in a previous course investigated the possibility of repairing buggy and insecure code with known good code, so I have a great appreciation for the work involved in this. My previous research on this topic was more syntactic, however, so I like that the authors did a different approach by doing a semantic search. I think it's an excellent idea to focus on what a piece of code does rather than how it's expressed, since that can vary depending on the programmer and it's difficult to account for such variations using traditional approaches like abstract syntax trees.

That being said, I really doubt the ability of this approach when the complexity of the code scales up. I think a lot more research needs to be done before this can reasonably be used in production code.

As a future research idea, I'd suggest looking into how more complex code and more realistic production code would fare and how the approach could be improved to account for those.

Current challenges in automatic software repair

I think the paper did a good job giving an overview of GenProg and the related work being done in the area of automatic software repair. I had never heard of GenProg before and when I read the word genetic programming I was at first completely lost, and unlike a lot of papers, I felt like this one did explain it reasonably well.

I would have liked to see more comparisons with other automatic software repair approaches and see how GenProg truly matched up.

As a future research idea I'd like to see more research into genetic programming and perhaps combining it with other automatic software repair approaches to see if can improve the results.

Is the Cure Worse Than the Disease? Overfitting in Automated Program Repair

I tend to enjoy papers when they cover topics or ideas that I had neglected to consider by myself so for that reason I enjoyed reading this paper. Now that I think about it, I think overfitting is definitely a huge concern, to the point that I can't believe I didn't consider it before! Even as humans writing code we tend to overcorrect for the simplest of problems, so it makes a lot of sense that any automatic program repair tool would be liable to do the same too.

As a future research idea, I'd like to see this topic applied to other automated tools as well. I'd like to see other automated tools compared to see if they go overboard too, and to see whether that affects the quality at all.