REACTION PAPER

The core concept which was presented involving ‘how to solve the Expression problem’ was well articulated given the fact that the criteria to define the expression problem was defined in terms of three independent points. The language, the domain and the approach were well presented as the 3D problem where the goal was the reach the plane where the solution was independent of all three. The speaker demonstrated the fact how the code has worked throughout the years by plotting it in 3D. That gave a clear picture as to what the final goal was. The productivity of the work was well highlighted with the fact that CoGen which was a step to reach EpCoGen was a useful tool which is used to replicate other research works. The speaker also used the comparative approach to differentiate the work from the ones which are quite popular like the ChatGPT. This is quite a practical way to connect with the audience, as that would be one the questions which many of the audiences might have had. ‘How is it different from ChatGPT’ which is known for solving problems. ChatGPT gives a solution which is one level lower than the solution given by the speaker i.e it repeats the Visitor Pattern. The speaker had already highlighted the problem with that approach previously. And the solution was CoGen which as the speaker rightly mentioned was not a Blackbox like the ChatGPT and something which can be controlled. The significance of research was definitely underlined by the fact.

The research could gain a niche in the Robotics industry where you keep adding new functionalities without disturbing the old ones. It would a waste of time and resource to test if the old functionalities would still work with the same efficiency even with the new functionalities added. Embedded software or software which is certified would be the right field where this work would be rightly appreciated. The speaker also goes on to give the example of Voyager spacecraft where the code is deployed remotely and might not be an ideal case to test all the previous functionalities. The rigidity of the specifications is what gives productivity in this case. The old code cannot be touched is what makes all the practical applications feasible. In the software industry like in the entertainment section in cars if a new function like the ‘reminder’ component is added, there would be a lot of testing so that it does not impact other modules like the Navigation or the Audio component. This might become obsolete with the current research if it is possible to reach the stage where it is independent of the language, domain and the approach.

The speaker also goes to explain the research methodology implemented such as modularity and also gives a parallel to it as it is more practical for humans to understand. Also, another important point was highlighted by the speaker where it is necessary to replicate others work to understand the core concerns in the field. The speaker also shows the timeline of the research or the solutions which have evolved over the years along with highlighting the fact that the problem continues to exist. One among them was defining the Visitor Pattern as a way to separate the algorithm from the object structure over which it operates, but the problem of changing the code continues to exists which was well explained by the speaker. The direction in which the research progressed was well reasoned which is quite intuitive. The speaker first explained the work and its application such was CoGen along with highlighting the loopholes which still existed in the work such as the problems with code generation such as the minor syntax errors. The temporary solution was to abstract the syntax, though different languages could have different ASTs. This paved the path for the speaker to connect to the next logical step which was to abstract the language, approach and even the application. The flow of thought was well organized and logical. Through this approach the speaker made it apparent that the next step would be find what was common across all programming languages and enter ‘Meta programming’, where the goal was to develop AST level API which is language independent.

Flowcharts were used to effectively communicate the idea which was presented; however, it did not do justice to the idea being presented. The code lines were displayed which were right on point in communicating the fact that it was a complex procedure. The flow of the solution which was suggested could have been better explained with a animation or a detailed flowchart. There was not enough time to process the code being presented and was useful to just provide an overview. Generating the code for *mi* and then evolving the model to generate the code for *mi+1* without changing the existing source code and also not copy the non-trivial code was well presented with a diagram. The speaker mentioned the two domains such as Mathematics and geometric shapes could also make one wonder if this is the right approach to other domains such as Machine Learning where certain languages such as Python is the go-to language.

‘Meta programming’ seems to solve most of the problems which were highlighted by the speaker and the next go-to thing in the software industry. Whenever a new language is developed it takes time for the supportive eco-system to develop and then adapted by programmers. This would not be a concern anymore as Meta programming would act as a transparent bridge between the language and the programmers. Even from the application point of view, it would the idea which will be highlight of the day and not the platform. This would definitely improve the productivity in the long run.