**Facilitating Programming Problem Formulations with a Decision Making Model**

Keywords: multi-criteria decision analysis, web-foraging, software design tradeoffs

**Introduction**

Programming is a cognitively demanding task, often requiring actions such as searching for solutions online and then making sense of and deciding between all the available alternatives. Previous investigations has shown that programming is a highly social process -- programmers facing a problem reaches out to their social networks (via face-to-face communication) as a first line of inquiry [1]. As the fastest Q&A platform for social computing [2], Stack Overflow provides an online alternative of crowdsourcing information from a social network on the global scale. However, the rapid growth of the Q&A site for programmers has caused a substantial surge in the number of unanswered questions [4], demonstrating a need for support in the question-formation process. Meanwhile, another emergent theme among software engineers is the struggle with formulating and maintaining mental models for code [1]. Leveraging an available corpus of successful (answered) questions on Stack Overflow, I intend to use multi-criteria decision analysis to help programmers formulate and state their problem objectives and specifications.

**Related Work**

State-of-the-art classification approaches have uncovered a set factors found in successful Stack Overflow questions [3, 4], these include the presence of code snippets, context-building via links and citations, post length as well as the expression of an affective state. Utilizing these drivers of success, I am currently working on constructing a generalizable rubric for identifying successful questions on Stack Overflow and validating the result by testing it on a sample containing both answered and unanswered questions.

My previous qualitative analysis has shown that a significant portion (51%) of sampled questions from Stack Overflow can be visually represented by the comparison table. In particular, any problem can be broken down and organized into the comparison table model when it contains multiple criteria (requirements or characteristics that helps measure utility of available options for a certain objective) and alternatives (potential solutions to the stated problem). These components directly map to the parts of the multi-criteria decision making (MCDM) model, which is preferred over the standard decision-making model due to having the benefit of accounting for potentially conflicting objectives. Hence, I will utilize the standard steps of forming a MCDM problem to guide the programmer’s process in constructing their question. One significant advancement will be the presence of recommended criteria: after amassing a corpus of formulated tables, a selection of criteria from similar questions can be detected when the asker begins composing their question. This feature will streamline the contextualization process since novice programmers are often unfamiliar with the environments of the problems that they inquire about.

**Methodology**

The steps toward developing support for developers’ question-formulation process on Stack Overflow can be broken down to four major stages: data collection, automation, and tool building and validation through usability testing.

Stage 1: Begin with a small sample of tables authored by researchers. Their validity can be verified via tasks on Amazon Mechanical Turk (after presenting a description of the model in the assignment instructions) . As a second part of the task, Turkers will be asked to convert an sample set of questions into the MCDM model, since they already have working knowledge of its components.

Stage 2: Since it is temporally unrealistic to expect the conversion of a diverse set of questions into comparison tables with sheer power of humans, we can adopt an automated approach of developing the set of comparison table. Using the initial set of crowdsourced tables from AMT, the expertise of faculty on natural language processing, and access to computing resources such as XSEDE, a model can be trained to automatically perform this task at scale.

Stage 3: After obtaining this sample of tables from a diverse variety of questions, I will build an assistive tool that locates and suggests criteria from the tables of similar questions when a programmer formulates their question on Stack Overflow. In addition to selection, the tool will also allow the forager to score the level the relevance of each selected criteria.

Stage 4: Finally, to evaluate the success of the tool, usability tests can be run to determine whether the tool helps to significantly improve the quality of questions formulated.

**Intellectual Merit / Broader Impacts**

A significant portion of questions are unanswered due to the lack of provided context. In fact, I hypothesize that the preference for face-to-face communication between programmers is due to the need for help with forming coherent and contextualized problem statements rather than the direct lack of knowledge – such an effect would explain the effectiveness of the rubber duck debugging technique. The proposed tool would help programmer build context in real-time by rating the importance (or irrelevance) of criteria recommended based on similar questions.

This table-building process adds specificity to the asker’s problem formulation, and helps the programmer construct a mental model by requiring them document their decision making process when identifying the options and criteria, as well as defining weights for each criterion. Furthermore, the assembled table will have the dual benefit of accelerating subsequent programmers’ sensemaking of the same problem by providing relevant contextual information (via the criteria and their respective weightings) and contributing to formulations of similar questions in the future. The results of these investigation and impacts of the developed tool will help advance our understanding of the knowledge sharing in the context of programming and make the Stack Overflow platform more accessible for a younger and broader community of software engineers.

**References**

[1] LaToza, T. D., Venolia, G., & DeLine, R. (2006, May). Maintaining mental models: a study of developer work habits. In *Proceedings of the 28th international conference on Software engineering* (pp. 492-501). ACM.

[2] Mamykina, L., Manoim, B., Mittal, M., Hripcsak, G., & Hartmann, B. (2011, May). Design lessons from the fastest q&a site in the west. In *Proceedings of the SIGCHI conference on Human factors in computing systems* (pp. 2857-2866). ACM.

[3] Calefato, F., Lanubile, F., Marasciulo, M. C., & Novielli, N. (2015, May). Mining successful answers in stack overflow. In Proceedings of the 12th Working Conference on Mining Software Repositories (pp. 430-433). IEEE Press.

[4] Asaduzzaman, M., Mashiyat, A. S., Roy, C. K., & Schneider, K. A. (2013, May). Answering questions about unanswered questions of stack overflow. In *Proceedings of the 10th Working Conference on Mining Software Repositories* (pp. 97-100). IEEE Press.