SHF: Small: Does more *ADVICE* improve interactive Search-based Requirements Engineering?

Tim Menzies (PI) IEEE Fellow; and Sandeep Kuttal (co-PI), NC State

OVERVIEW

Software engineering (SE) has advanced software solutions, however, currently, we struggle and often fail, to understand the increasingly complex models that we are building. Enabling humans to explore all the important potential behaviors of a software model is an open and important issue. Unfortunately, there are substantial examples where human oversight missed important software properties.

To forge an effective partnership, humans and artificial intelligence (AI) need to understand each other's strengths and limitations. For example, interactive search-based SE tools (iSBSE) work well when taking advice from one person but have issues dealing with advice from large teams. Therefore, we propose a system to extend iSBSE with particle-swarm optimization and generative transformer models to handle teams; specifically: (1) debates and disagreements between team members; (2) team members with an established track record of offering good/bad advice; and (3) team members that (consciously or unconsciously) offer advice that leads to discriminatory models.

Our new system will monitor policy decisions made by AI or humans. To facilitate open science, all the scripts, models, and data used in this project will be stored on Github and be freely available to researchers and industrial practitioners.

INTELLECTUAL MERIT

The project proposes advanced *collaborative discovery* software for students, faculty, and industry researchers in software engineering, artificial intelligence, and human-computer interaction, to realize a solution that can transform complex decision-making for stakeholders.

The proposed research is original (none of the SAT problems in software engineering consider incorporating team disputes), transformative (potential to change decision making when designing software), and feasible (validated by user studies and preliminary designs). Specific aspects of the research that contribute to its intellectual merits are: (1) Identifying specific cues related to social and cognition; modeling these to facilitate the decision-making process of stakeholders; (2) Tight data collection, design, evaluation, and refinement cycles; (3) Diverse and focused empirical user studies will drive and refine our design; (4) Overall research approach can be applied to SAT problems to help to make decisions. The approaches involve identifying cues from individual humans and from logs, defining the inclusive model, verifying the results, create appropriate messages and interactions.

BROADER IMPACTS

NC State's Computer Science department has a long tradition of studying research issues related to gender bias, barriers faced by women, and methods for broadening participation in the context of software engineering. This work will continue that tradition. Curriculum and lecture notes of the graduate SE classes (taught by PI Menzies and co-PI Sandeep Kuttal) will be revised from this work. Funds from this work will be used to support students attending the Grace Hopper conference, and the Richard Tapia Celebration of Diversity in Computing. Furthermore, a (small) portion of this grant would be allocated to support Broadening Participation in Computing (BPC) work. PI Menzies and co-PI Kuttal are members of their department's Broadening Participation Committee (BPC) which actively seeks to: (a) Understand what factors make computer science more (or less) attractive to underrepresented groups; (b) Educate faculty, staff, and students on how different behaviors affect diversity, quality, and inclusiveness; (c) Increase the percent of students who identify as women, and (d) Evaluate the success of this BPC team in broadening participation.

KEYWORDS

Software Analytics, Requirements Engineering, Search-based Software Engineering

SHF: Small: Does more *ADVICE* improve interactive Search-based Requirements Engineering?

Tim Menzies (PI) IEEE Fellow; and Sandeep Kuttal (co-PI), NC State

1 Introduction

Software engineering (SE) can now solve complex problems. But those increasingly complex models are also increasingly hard to understand– making them harder to extend and maintain (and they may contain hidden bugs). For example, Figure 1 lists the the information needs of a university CS department [40]. This model is so large that, in our experience [72], humans daunted by its complexity. Even automated tools struggle to find choices that satisfy the goals of that model; e.g. the NSGA-II [26] multi-objective algorithm needs 1000s of seconds and 100,000s of evaluations to explore Figure 1 [72].

We conjecture that a combination of advice from *both* human *and* AI sources can find the important aspects of complex software models. To test this conjecture, we note that once we recognize (a) *what model* is being explored, and (b) *what team* is exploring it, then there exists ample (potentially) relevant *cloud information* available from the web about that team and model (see Table 1). Hence, we investigate:

> If models use team and cloud knowledge, will that make systems' decision-making better or worse?

This is an important question. Clearly, too much divergent advice from too many sources is counter-productive. Hence, we explore research questions like when does "enough " *ADVICE* become "too much"?

This research team already some initial results in this research direction. Our *ADVICE*-0 research prototype explored iSBSE models using individual human help to guide that search [67,68]. While *ADVICE*-0 lets *one* person explore a model, much research is required before that tool can support *teams*. Hence, this proposal offers a three-year plan to address those shortcomings via four new systems: *ADVICE*-1, *ADVICE*-2, *ADVICE*-3, and *ADVICE*-4. Each one of these systems will involve extensive user studies and addresses fundamental issues in model-based requirements engineering.

We note that this research team has the experience (quantitative algorithmic reasoning and human qualitative studies) needed to successfully accomplish this project. PI Menzies is working on fast algorithms for requirements engineering and has been for decades [27,78] and recently had success in finding "shot-cuts" that allow for the rapid processing of very large models [67,68,73,81]. Co-PI Kuttal has extensive expertise on tool building (e.g., [56,59,92]), conducting quantitative and qualitative studies (e.g., [54,71,99,108,111]), studying gender and expert based biases (e.g., [5,45,53,65]), and mining cloud information (e.g., [52,58,94,104]).

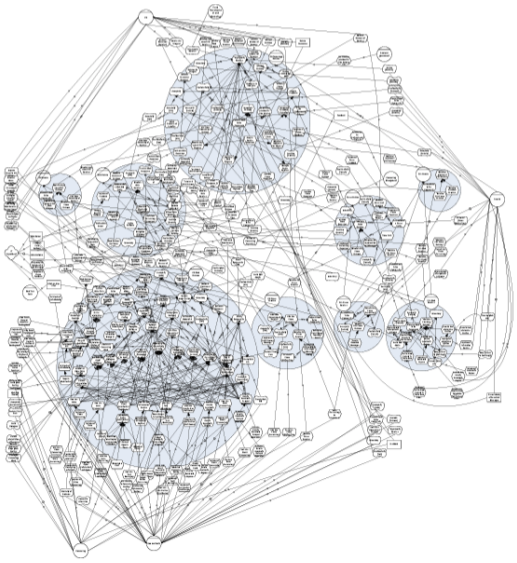
Given access to downloadable pre-trained language models (e.g. BERT), then a small list of words from a model can be expanded to a larger list of relevant terms [35] (which can seed subsequent searchers around the web).
Given knowledge of someone's user name on Stack Overflow or medium.com or other online forums, it is possible to access opinion pieces written by different people.
Given permission (from users) to access browser history, we can see where a human has *lingered* longer while browsing the web. Under the assumption that a longer inspection of a (say) Stack Overflow page means more interest in that material, then we can also use *lingering* to learn what other opinions they also care about.
For project under version control, we can access who has commented/revised what parts of the design documents.
Given the version control system with comments, plus some systems for registering "votes" (e.g. emoticons for thumbs up and thumbs down), we can access *current* and *past* decisions of a team member.
Given sentiment analysis tools [85] running on all the text seen in the last few points, it would be possible to learn the "hot button" issues for different team members.
Given natural language tools like Latent Dirichlet Allocation [7] or word2vec [79] or BERT [18], it is possible to find clusters that characterize the text generated by our team members, then find which clusters contain the words most liked/disliked by our team members.

Table 1: Cloud knowledge.

2 Background

Enabling humans to explore all the important potential behaviors of a software model is an open and important issue. In "Flaws of policies of requiring human oversight" [33], Ben Green notes that many recent policies require humans-in-the-loop to review or audit decisions from software models. E.g. the manual of the (in)famous COMPAS model (see Table 2) notes the algorithm can make mistakes and advises that "staff should be encouraged to use their professional judgment and override the computed risk as appropriate" [84].

Figure 1: 351 options for CS department services, inked via 510 edges (i* format [17]).



Cognitive theory [102] tells us that humans use heuristic "cues" that lead them to the most important parts of a model before moving on to their next task. But when humans review models, they can miss important details. Such cues are essential if humans are to tackle their busy workloads. That said, using cues can introduce errors: *...people (including experts) are susceptible to "automation bias" (involving) omission errors—failing to take action because the automated system did not provide an alert—and commission error* [33]. This means that oversight policies can lead to the reverse of their desired effect by *"legitimizing the use of faulty and controversial algorithms without addressing (their fundamental issues"* [33].

Unfortunately, there are substantial examples where human oversight missed important software properties (see Table 2). For example, it took years to realize that COMPAS had an alarming difference in the false alarm rates for black and white defendants [21]. That difference is a bug (since it can be fixed– see PI Menzies' FSE'21 paper [22] that weights training examples to reduce COMPAS's false alarm delta, while maintaining the same levels of recall on actual recidivism).

To forge an effective partnership, humans and artificial intelligence (AI) need to understand each other's strengths and limitations. The software can explore a very large space, on pre-determined criteria. Humans can offer novel insight, but only over a small number of examples. We conjecture, that when combined, both can find better solutions than if either worked separately. For example, working with humans, our *ADVICE*-0 software tool [67, 68] explores trillions of options to find a few, most important, decisions that give the best results from a model. *ADVICE*-0 takes advice from humans– but only sparingly. In studies with a dozen SE models *ADVICE*-0 cued into solutions within 1% to 3% of optimum after asking humans 10 to 100 questions (and prior state-of-the-art tool [9], which used stochastic evolutionary algorithms, needed 1,000 to 100,000 questions to get equivalent results). Interestingly, the decisions found by *ADVICE*-0 out-performed state-of-the-art optimisers such as FLASH, HYPEROPT and OPTUNA [8, 12, 81]. We conjecture that those other optimisers performed worse since they used a somewhat uninformed search based on random mutations. On the other hand, *ADVICE*-0 works well because (a) it combined

The COMPAS recidivism models labels black defendants as future criminals at twice the rate as whites [2].
Widely-used face recognition software predicting gender & age, has a much higher error rate for dark-skinned women compared to light-skinned men [3].
Amazon's software for same-day delivery to prime users became biased against black neighborhoods [1].
Google Translate has gender bias. "She is an engineer, He is a nurse" translated to Turkish then back to English gives "He is an engineer, She is a nurse" [19].

Table 2: Example biases seen in software decisions. For more, see Rudin [93], Nobel [83], Gebru [31].

insights from *both* human and algorithmic sources and (b) using the data mining operators defined later in this proposal (see Table 3), it reflected the shape of the data before deciding where to go next.

While a promising first step, *ADVICE*-0 has many limitations: (a) it could not take advice from the teams; (b) it only took advice from humans, ignoring all the background knowledge available from the cloud; (c) it never checked for mistaken advice; and (d) it never checked itself for bias or discriminatory consequences (hence, in its current form, *ADVICE*-0 cannot address issues like those seen in Table 2).

Hence, as shown (at right), we propose a three year plan to build four extensions to *ADVICE*-0 called *ADVICE*-1, *ADVICE*-2, *ADVICE*-3, and *ADVICE*-4 that address these problems (a),(b),(c), and (d). All these tools will be freely available on Github under open source licenses. One reason to fund this proposal is that even if some of our steps fail, we can still produce useful results. For example, even if we do not complete *ADVICE*-4, at the very least this research would result in a new version *ADVICE*-0+, augmented with (e.g.) all the bias recognition tools we will develop for *ADVICE*-4.

	Year	Functionality
ADVICE-0	1	no team, no cloud
ADVICE-1	1	+ a team of stakeholders
ADVICE-2	2	+ cloud knowledge
ADVICE-3	2	+ verification
ADVICE-4	3	+ bias mitigation

2.1 Terminology and Scope

- By *cloud knowledge*, we mean the information in Table 1 comprising all the extra web content that can better inform us about the context of some piece of software.
- By *stakeholders* we mean anyone who cares about the software design such as managers, developers, U/IX designers, lawyers (auditing a design for discrimination), and members of community groups trying to understand and improve what is going on in a model.
- By *team*, we mean a group of up to a dozen *stakeholders* arguing about some software, or its design. *ADVICE*-0 refers to a system with *no* team or cloud knowledge.
- This proposal is focused on *requirements engineering* (RE). RE used to happen before analysis, design, coding, and testing (and for safety-critical applications, there is still a pressing need for this to occur before coding starts). But in the age of DevOps and Autonomous/self-adaptive systems and Software 2.0, requirements can be explored many times in a software project [90, 95]. Hence, after Bencomo et al. [11], we say *"Requirements engineering is any discussion about what to build and how to trade-off competing costs/benefits. It can happen before, during, or after runtime [11]."*.
- *Search-based RE* seeks optimal or near optimal solutions in a search space of candidate solutions, guided by a fitness function that distinguishes better and worse solutions [36].
- *Interactive Search-based RE* lets stakeholders interactively offer advice to better guide that search [9].
- We will explore *model-based* RE, not "green-field" discussions. That is to say, participants in our experiments will NOT start with some blank sheet of paper. Rather, they will change some pre-existing knowledge in some models. We make this choice since, in this age of the mash-up, the software is rarely built without the consideration of some set of background constraints such as pre-existing requirements, legislative or corporate policies, prior design decisions, etc.
- In this proposal, we will focus on *serial RE discussions* where some design artifact (the model) is being passed along a circle of stakeholders who may update the model before passing it down the circle (stopping when the model traverses the entire circle without revisions or comments). This research project will form foundations for exploring "parallel RE design discussions" where N people debate the same version of a model at the same time (future work).
- While we mostly focus on requirements engineering, there is an obvious extension of this work to *minimal test suite* generation. This will be explored in one of our latter research questions.

2.2 Example (and our prior work with the *ADVICE*-0 system)

This work assumes the existence of a model; i.e. a space of options that discusses where it is possible (or impossible) to combine certain items. Examples of such formats include the i* notation used in Figure 1 and the *feature model* format [44] used in Figure 2. Figure 2 is Mendonca et al.'s representation [75] of

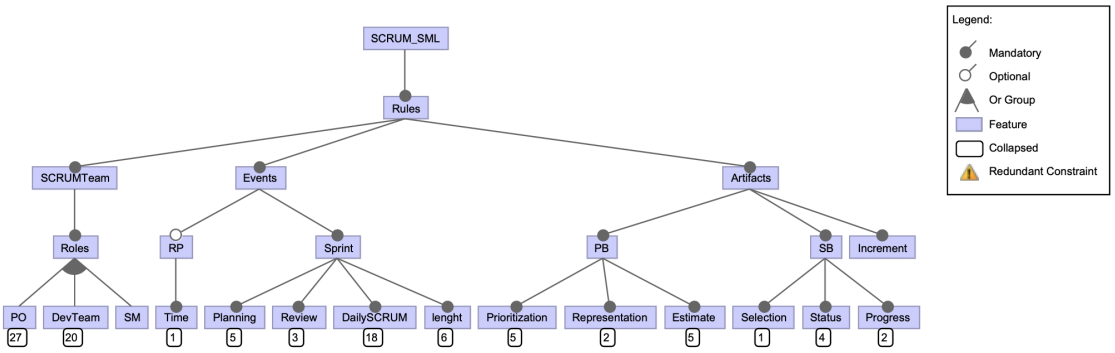


Figure 2: SCRUM Feature Model. Numbers on leaves show #constraints in that part of the model. Not shown here, for space reasons, are "cross-tree constraints" that connect choices in different sub-trees.

SCRUM (suggestions on how to run an agile project). The model has 128 project management options and 250+ constraints (e.g., if sprints last two weeks, then each individual task must take less than 10 days of programming) [75]. With the constraints, less than 2% of choices are acceptable. Each choice contains:

- An estimated development effort;
- A purchasing cost (which is non-zero for features associated with third-party libraries);
- Some number of defects seen in past developments;
- A "success" number that is incremented if this node had been used in some prior successful project.

In those experiments, we selected options while trying to *minimize* the sum of (i) cost and (ii) effort and (iii) defects while *maximizing* the sum of the (iv) features delivered and the (v) the "success" score. To say the least, it is very hard for humans to find solutions that satisfy the constraints while optimizing for these five goals within a space of 2^{128}*2% possibilities [68].

When reasoning over many constraints, AI tools are useful. For example, the PicoSAT [15], SAT solver can find tens of millions of satisfying solutions to Figure 2. But now there is a new problem: **too many solutions**. When tools like can PICOSAT can find millions of solutions, interactive search-based requirements engineering can use human preferences to find just a few cues that select for good solutions acceptable to most stakeholders. *ADVICE*-0 was a prototype to see how well we could find the cues, while asking participants the least number of questions [68]. *ADVICE*-0 applied the data mining operators of Table 3 to find cues that both (a) selected for good solutions most acceptable to participants that (b) ruled out the most number of the remaining options (so that there are very few questions needed to be asked next). *ADVICE*-0 is a semi-unsupervised [24] active learner [100] that decides between options $X = \{x_1, x_2, ..\}$. Options are scored by a human f on goals y_i, so

$$y_i = f(x_i) \tag{1}$$

where x are (e.g.) 128 decisions within the model of Figure 2 and y are multiple objectives such as "Minimize size of team" or "Deliver more product sooner". *ADVICE* assumes that is **expensive** to call f (since each such calls means pestering a human for an expert opinion) but **cheap** to find many x_i vectors (e.g. random selection over 10 binary options generates $|X| \leq 2^{10} = 1024$ options). As discussed in Table 3, *ADVICE*-0 applied a range of data miner operators[1] to (a) divide up the x space, then (b) ask the user questions about the most important "cues"; i.e. the fewest number x decisions that prunes most options.

Just to highlight the benefits of our approach, we note that when *ADVICE*-0 asks a question, we *do not* mean users are asked to pick between two lists of the 128 features in Figure 2 (since users would

[1] Random projections to recursively bi-cluster the data. Entropy measures to assess which clusters should be explored first. Semi-supervised learning to cluster the space, then ask only a few questions per cluster. Feature selection to select which attributes are most informative. For details, see Table 3.

ADVICE-0 builds a binary tree of clusters from X examples. The *diversity* D parts of that tree is the sum of the diversity of the x_i columns in that subset (for numerics and symbolics, this is variance and entropy, respectively).

The diversity of subtrees t_1, t_2 is t_1, t_2 is $D(t_1) + D(t_2)$. *ADVICE*-0 asks questions about *most divisive* splits; i.e.the largest splits with lowest diversity.

ADVICE-0 asks about features that pass *feature selection*; i.e. those most distinguish two sibling splits.

After asking the user what x_i values they prefer, *ADVICE*-0 deletes the less preferred half, then loops to the next most divisive split (stopping s when $N = |X|$ options has been reduced to $n = \sqrt{N}$). Importantly, it does so with zero calls to f (though it has had to ask the user a few questions about some x_i features).

These n values are further pruned by a post-processor that (a) finds two distant a, b vectors; (b) runs f to find the a and b; (b) recurses on half the data nearest the best of a and b value (stopping when at $\sqrt{n}$ of the data).

To decide which of a and b, we apply the Zitzler [112] continuous domination predictor to the y values of those vectors. This predicate favors a over b model if jumping from a "loses" most:

$$worse(a,b) = loss(A,B) > loss(a,b) \quad | \quad loss(a,b) = \sum_{j=1}^{n} -e^{\Delta(j,a,b,n)}/n \quad | \quad \Delta(j,a,y,n) = w_j(y_j^a - y_j^b)/n \tag{2}$$

where "n" is the number of objectives and $w_j \in \{-1, 1\}$ depending on whether we seek to maximize goal x_j and $o_{j,A}, o_{j,B}$ are the scores seen for objective o_j for A, B, respectively. Zitzler preferred "boolean domination" (one thing is better than another if it is no worse on any criteria and better on at least one criterion) since, it is known that boolean domination can fail for three or more goals [97, 109].

Post-processor needs $2log_2(n)$ calls to find $\sqrt{\sqrt{N}}$ options that survived the tree pruning *and* the post-processing.

Table 3: *ADVICE*-0 automatically finds cues that guide to better solutions. Later in this proposal, we will assess the value of (a) automatically finding cues versus (b) surveying humans to find their cues.

quickly tire of those kinds of questions). Rather, when we say "*ADVICE*-0 asks the users for a question", that question asks for preferences amongst the *fewest* features that *most* distinguishes clusters within the current space of options. In this way, answers to each question can quickly wipe out large sections of the search space, thus reducing the number of subsequent questions we will ever need to ask.

As said above, in studies [68] with a dozen SE models, *ADVICE*-0 cued into solutions within 1% to 3% of optimum after asking humans 10 to 100 questions. The decisions found by *ADVICE*-0 outperformed state-of-the-art optimizers such as FLASH, HYPEROPT and OPTUNA [8, 12, 81]. We conjecture that those other optimizers before performed worse since they used a somewhat uninformed search based on random mutations. On the other hand, *ADVICE*-0 works so well since (a) it combined insights from *both* human and algorithmic sources and (b) using the data mining operators defined later in this proposal (see Table 3), it reflected the shape of the data before deciding where to go next.

3 How to do it, better? (Design of our Proposed System: Challenges and Solutions)

As stated in our introduction, while a promising first step, *ADVICE*-0 has many limitations: (a) it could not take advice from teams of participants; (b) it only took advice from humans, ignoring all the knowledge available from the cloud; (c) it never checked for mistaken advice; and (d) it never checked itself for bias or discriminatory consequences. We assert that these are not just problems with *ADVICE*-0 but also problems with many other interactive search-based RE tool.

The rest of this proposal discusses ways to address problems (a),(b),(c), and (d). In summary:

- Instead of making a single conclusion, *ADVICE*-1 will use particle swarm optimization (described below) to give each stakeholder N particles, all of which will debate where to find good solutions.
- Instead of using just the raw model, *ADVICE*-2 will search over an extended space learned from Table 1.
- As the particles search, we will sometimes take advice from human participants. Instead of accepting that advice uncritically, *ADVICE*-3 will test and weight that advice according to what has been useful in the past, (and rejecting advice that leads to anomalies).
- *ADVICE*-4 will find and fix inappropriate performance differences effecting different categories of users

(thus reducing discriminatory effects of these models).

As said above, an important aspect of this research plan is its *survivability*. Given the multiple goals of *ADVICE*-1 .. *ADVICE*-4, if we fail on any one goal, we can still continue to succeed on the others.

3.1 *ADVICE*-1: Adding Team-based Reasoning

At sunset in Siberia, flocks of starlings fly in beautiful patterns called a murmuration. This complex group behavior has no centralized controller. Rather, it results from many local starling adjusting their position via feedback from the neighbors.

Inspired by these starlings, researchers in *particle swam optimization* (PSO) [46] implemented optimization as a set of candidates "flying around" (i.e. being mutated) a shared space. One advantage of this scheme is that it lends itself to a group performing model maintenance. In PSO, particles do not just arrive at some location and stop. Instead, each particle is like a helicopter buzzing around the landscape. New model conditions are like a breeze that pushes the particles some distance across the decision space in a model. Our particles then must make new decisions as they negotiate their way back to their preferred positions.

We argue that PSO is a natural method for negotiating shared solutions. Particle velocity is controlled by *inertial*, *cognitive*, and *social* weights w, ϕ_p, ϕ_g (where p, g are short for "personnel" and "group"). In a swam of "particles" p "flying around" (i.e. being mutated across a set of vectors), then:

- The w *interia* factor pushes p_i along the current direction.
- While the *cognitive* and *social* weightings ϕ_p, ϕ_g pulls p_i towards (a) the best solution found this particle or (b) the best solution found by the entire swarm of particles.
- w, ϕ_p, ϕ_g serve to nudge p_i to a new direction.

PSO's particles find a balance between past decisions (w), the preferences of one explorer (ϕ_p), and the preferences made by the team (ϕ_g). In our scheme, stakeholders gets multiple particles, initialized to:

- An initial random position within the space of options.
- Or, if the crowd knowledge is available (see next section), positioned admist the crowd knowledge collected for one stakeholder.

Initial particle velocity is set according to how far, and in what direction, are the ideal answers for each individual (the further you start from your ideal, the more aggressively you jump towards it).

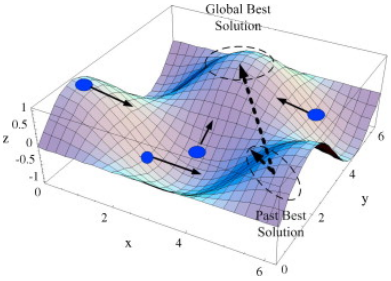


Figure 3: Particle Swarm Optimization.

To prepare for this proposal, we built a simple PSO prototype for Figure 2. That prototype used 1000s to 100,000s of evaluations in order to explore those 2^{120} options. If each evaluation meant one question to the stakeholders, then this would confuse and overwhelm them.

To reduce that problem, we take inspiration from a 2021 paper by Mai et al. [69]. They used PSO with the evaluation oracle replaced with fuzzy c-means clustering (FCM)[2]. When FCM needs labels, it computes them from a weighted sum of the labeled examples in nearby clusters. While an exemplary approach in many respects Mai et al.'s design still makes 100s to 1000s (or more) queries to some oracle. To make PSO palatable for humans, we adapt Mai et al., replacing fuzzy c-means with the methods of Table 3 (i.e. *ADVICE*-0 will be a small sub-routine inside *ADVICE*-1 .. *ADVICE*-4). In this way, humans would only be bothered for their opinions on a small number of most critical questions.

[2] In FCM, examples may appear in multiple clusters. Membership probabilities are computed by iteratively updating a matrix of membership values using values from the last iteration– then repeating until convergence is reached.

3.2 *ADVICE*-2 = *ADVICE*-1 + Additional Knowledge from the Cloud

With *ADVICE*-0 [67, 68] we found that many preferences about x decisions are "soft"; i.e. open to negotiation; e.g. a developer might say "I prefer C, but I can code in C++ if you need me too.". To handle soft goals, we propose adding a new term x preference term ϕ_x to the set w, ϕ_p, ϕ_g that controls PSO. The PSO we propose has different particles assigned to different stakeholders; and an extra term ϕ_x that nudges particles towards/away from things that this stakeholder likes/dislikes.

More specifically, after creating a large cloud of randomly selected x decisions, *ADVICE* will use the electronic footprint to annotate some parts of that cloud as "great place come try over here!" or "terrible place! please try to avoid!". This footprint can be created two ways:

- *Text mining of existing terms* generating summaries of information from Table 1. Co-PI Kuttal have used information sources such as GitHub, Stack Overflow, and App Stores to mine the technical and social skills of developers [52, 94], study migration behavior across platforms [104], compare the behavior of app users (compared to developers) [51], and model foraging behavior of developers [89, 99, 111].
- *Expanding existing sets of terminology.* Recently, researchers have applied large pre-trained language models to take a set of domain terms, then expand that to a large one using associations offered by the language model. For example, Fu et al. [29] recently reported success in improving predictions using BERT to expand the usecase stories vocab [18]. Co-PI Kuttal used transformer-based language models, specifically, BERT, GPT2, and XLNet, to classify the intent of developer conversations [5, 41, 86].

To use those terms here, we would weight terminology from Table 1, by how much humans like (or loathe) each term. Lin et al. [62] says *sentiment analysis* finds objective states and subjective opinions reported in sentences (sentiment analysis can classify customers' written opinions as negative, neutral, or positive). For example, the footprint of stakeholders concerned with issues like Table 1 might make express a positive sentiment towards references about (a) specific social groups (women, veterans) or (b) verification technology that evaluates a system.

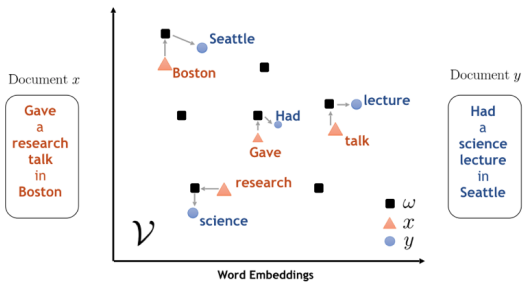


Figure 4: Adjusting objective space. From IBM: blogs/research/2018/11/word-movers-embedding/. More advanced kind of reasoning include systems like BERT [18].

To ensure a useful overlap between the vocabulary of the model and the vocabulary of cloud artifacts, we need synonym discovery. Such natural language synonym discovery [18] is illustrated in Figure 4. Some vector space is created with weights such similar terms will appear close to each other in the inferred vector space. Synonymn discovery can fail if a user's footprint is too far removed from the model being explored. That said, synonym discovery should work well in our domain where teams uses tools like GitHub to store and comment on work products. In that context, the electronic footprint of some user will be all the text and comments stored in GitHub *about this model*– in which case, synonym discovery should be very effective.

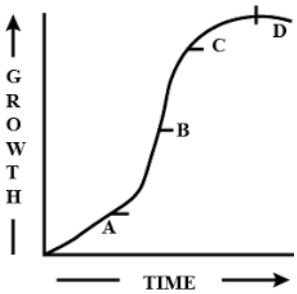
3.3 *ADVICE*-3 = *ADVICE*-2 + Advice Verification Tools

Given so many team members and so many knowledge sources, then we should expect that some team members and some knowledge sources are more useful than others. Using *meta-patterns of fault-less or fault-full behavior*, we need to distinguish and manage (a) the advice that is most helpful; and (b) the advice that most hurt our ability to satisfy more goals.

One such meta-pattern is *good advice needs less revision*. Suppose we have a version tracking system (e.g. GitHub) that watches model revisions proposed by a group of stakeholders. Suppose further those stakeholders are using the knowledge sources from Table 1, over several weeks or months. Those stakeholders and/or those knowledge sources can now be ranked by how often their advice is subsequently revised. Under the assumption that good advice is revised less, we can weigh our PSO explorations to

favor regions with "good" knowledge sources

Another such meta-pattern is *bad advice means more performance variance*. In a recent TSE'21 article [110], PI Menzies and his student Zhe Yu [110] found they could apply growth curve mathematics [20] to interactive search-based methods. At each "time tick", humans make some decisions and the performance scores change (hopefully, it improves). As shown at right, early on (at point "A"), PI Menzies and Yu were able to extrapolate forward the expected future shape of the graph. Such extrapolations fail when bad advice throws a project off course. In that case, a repair action could be to *replay* everything from the nearest point where the extrapolations were predicting accurately. Here, by "replay" we mean reset the model back to some early point *but keep all the answers from users seen since that point*. Then apply our inference tools to see what happens if we change the answer at the early point, then try to reapply the answers seen subsequently.



Potentially, there are many other meta-patterns such as *bad advice comes very quickly* or *good advice explores more options before deciding*. We plan extensive observation of our experimental participants to identify which meta-patterns are most useful.

3.4 *ADVICE*-4 = *ADVICE*-3 + Bias Mitigation

Here, we are talking about how to manage problems like those seen in Table 2.

To manage bias, first, we have to measure it. A recent literature review by PI Menzies and his student Suvodeep Majumder [70] lists current metrics seen in the SE literature that report performance variance between test cases from different social groups. Many of those are for categorical values (and our y-space objectives will be numeric), so we will adapt those too (e.g.) report the variance reduction if we jump to specific social groups (and the *more* variance reduction means *more* bias).

Now that we can measure bias, we must mitigate it. Four such mitigation methods are:

- *Personnel balancing*: As to *personnel balancing*, Leavey [60], Nobel [83] and Gebru [31] warn that a lack of diversity in a design team leads to designs that discriminate against particular sections of society. Co-PI Kuttal in her lab studies found that balancing helps in removing any gender and expert biases [5,45,53,65,91]. Issues like that motivate us to explore hybrid methods that combine personnel balancing with other methods:
- *Extra context:* Recall that that *ADVICE*2 expands model terminology using Table 1. Our experiments will check how bias is effected by increasing Table 1 information. Co-PI Kuttal analyzing the gender data found that BERT model needs extra context [5] while SVM models can use a gender feature [91].
- *Hyperparameter optimizers* are algorithms that find settings for data miners that improve predictively performance. As showed in our FSE'20 paper [23], that process can also be used to select models whose output decreases disparities between social groups (e.g. such as those reported in our introduction for COMPAS). Accordingly, for *ADVICE*-4, we would revisit that FSE'20 work to find even better Hyperparameter optimizers for PSO that remove even more disparity between social groupings.
- *Sample balancing* mitigates bias arises from an imbalanced sampling of social groupings by adjusting the ratios of those groupings (within the sample used to build a model). At FSE'21 PI Menzies and his student Joymallya Chakraborty earned a distinguished paper award [22] for a system that extrapolated examples within under-represented social groups within the training data[3]. Prior to that work, it was feared that repairing bias also meant damaging predictive efficacy[4] But at FSE'21, we showed that it was possible to maintain predictive prowess while, at the same time, reducing bias. Sample balancing could also be applied to *ADVICE*-4. We discussed the above methods for *ADVICE*'s PSO particles to take advice on where they should travel next. Our FSE'21 data balancing operators from PI Menzies and Chakraborty could also be applied to select better directions for our PSO particles.

[3] Implementation note: that system adjusted frequencies in the *training* data and **NOT** the *test* data.

[4] In 2017, Berk et al. [13] said "It is impossible to achieve fairness and high performance simultaneously (except in trivial cases".

4 Methods

This section explains the methods applied to investigate research questions. We will explore our models when stakeholders (participants) are asked occasional questions that guide the search process. *ADVICE* will seek compromise positions that satisfy, as far as possible, most goals.

Initially, small *lab studies* will be conducted to understand developers' decisions in controlled environments. We will work with teams of size $N = 1$, then move to larger teams for years two and three. After these smaller scale lab studies will come much larger scale *case studies*. For our *lab studies* we will use university students and for our *case studies*, we will use teams recruited from professional software organizations and job marketplace venues like Mechanical Turk. Note that Co-PI Kuttal is particularly skilled with such lab studies and case studies (e.g., [45, 66]).

TASKS: Our lab studies will explore small changes to an existing model. Participants will be given various tasks such as (a) seek choices that best optimize the y goals of our models; (b) find faults that we have seeded (where "fault" in this case means violations of pre-defined policy that exists in a paper version of the model); (c) review models for any missing requirements; (d) extend the model to handle new requirements. As to accessing models, there is a large supply of SE-related models in the i* format of Figure 1 or the feature model format of Figure 2. Table 4 lists some of the i* models we currently possess (and for more, we will do a literature review of the i* RE literature). As to feature models Figure 2, the original 1990 paper on feature models [44] has been used widely (it has 5,535 citations as of May 31, 2022). Hence, we can access feature models for a very large number of SE tasks[5] and, on the web, we can access 100s of large models of this format[6] has feature models of design of electronic shopping carts, different software security authentication schemes, operating system designs,etc..

Table 4: i* Models from [72].

Model	Nodes	Edges
Services, see Figure 2.	351	510
Counselling	350	470
Marketing	326	422
Management	206	239
ITDepartment	126	162
Kids&Youth	81	81
IT Modernization	53	57

WORKFLOW: As explained in our introduction, we use serial RE discussions where some design artifact (the model) is being passed along a circle of stakeholders who may update the model before passing it down the circle (stopping when the model traverses the entire circle without revisions of comments).

This workflow will make it convenient to conduct large scale studies where we invite industrial practitioners (via email) to participate in our studies. Co-PI Kuttal has much experience with this kind of study [111].

PARTICIPANTS: Initially, we will work with student groups (year1). Subsequently, we will broaden our user base to industrial practitioners. As to accessing student groups, At NC State, Co-PI Kuttal and PI Menzies both teach large SE classes (graduate and undergraduate). After obtaining appropriate Investigator Review Board (IRB) clearances from the NC STATE IRB committee, the studies will use teams comprising stakeholders with varying:

- Familiarity with the models being examined (as measured by pre-study questionnaires);
- Gender, racial, and other mixtures.

INTERFACE: The design space for the ADVICE interface will be iteratively improved based on evaluation from multiple stakeholders of various expertise (novice and professionals) and genders. Co-PI Kuttal has created human-centric tools for developers using user-centered design and evaluation approaches. She have created tools that support programmer creativity [50] and exploratory programming behavior [42, 43], debugging web-based distributed programming [55–57], problem-solving techniques [42, 43], gender-specific behavior [5, 45, 65], socio-technical skills [6, 52, 58, 94, 99, 104, 108, 111] and communication styles [50]. The final version or deployment version will be provided with a sophisticated interface that

[5] Configuring LINUX kernels [96]; software reuse [4, 74], software traceability [10] software project management [68], defect prediction [14], and design pattern discovery (i.e. looking for repeated patterns within part of different designs [106]), etc.

[6] at github: marcilio/splot-research/tree/master/SPLOT-Research/WebContent/models

allows explaining the decisions of the ADVICE. Further, it will also give suggestions based on Cloud and teams during the decision-making process of stakeholders. The explanation and suggestions will follow Shneiderman's guidelines [101] and Neilsen's heuristics [82]. Gender issues leading to biases will be explored and mitigated using GenderMag [25, 34], which helps find and fix gender-related issues in problem-solving software and increase gender inclusiveness.

METHODOLOGY: For data collection, the think-aloud method [61,98] will be used. Participants will vocalize their thoughts and feelings as they perform their tasks. Participants in a control group will perform tasks using appropriate state-of-the-art optimizers suitable for comparison with automated version of *ADVICE*. Based on our literature review [68] (as of 2021) FLASH and HYPEROPT and OPTUNA [8,12,81] will be utilized (we will update those comparison tools to reflect the ongoing state-of-the-art).

As in our past qualitative analyses (e.g., [50, 65]), we will triangulate (compare and attempt to refute) the results with interviews and surveys to understand their strengths. Retrospective interviews will be conducted to gain insights into participants' experiences and barriers encountered during decision-making with and without *ADVICE*. We will collect (1) Pre-surveys with standard questionnaires to collect demographics and familiarity with the domain of the model. (2) Pre- and post-surveys with standard questionnaires will be used to evaluate self-efficacy. (3) Post-surveys will be used to analyze the cognitive load using NASA Task Load Index [38] to measure and conduct a subjective mental workload (MWL) assessment.

DATA ANALYSIS: Video data, audio data, and screen interactions of developers will be collected as transcripts, which will be analyzed using a mix of qualitative and quantitative measures. Qualitative methods from Grounded Theory [32] (e.g., Corbin and Strauss variant [103]) will be used to analyze the transcripts. We will annotate points based on developer utterances to identify key concepts and phenomena via an iterative, open-coding process, especially for artifacts or experience-based cues when making decisions. This process will also be used to analyze decision-making behavior. Finally, thematic analysis [16] will be used to organize qualitative data into themes that relate back to the research questions. Non-parametric effect size and significance tests will be used for quantitative analysis.

Further to the above, some of our specific research questions require the collection of specific metrics:

- *Stakeholder category metrics :* One goal of this work is to better support model-based reasoning for different categories. Under some anonymization protocol, *participant diversity* will be collected (with categories such as age, gender, racial background, veteran status, accessible challenges, etc.)
- *Bias Metrics:* Our bias mitigation metrics were discussed in §3.4. Note that we would ensure we can report these metrics separately for all different stakeholder categories.
- *Initial and final divergence metrics:* For teams of size $N > 1$, we use PSO to measure initial and final divergence before and after inference (and, ideally, that difference decreases from initial to final). Recall from the above that stakeholders are assigned their own particles which move from some initial position to some final position. By measuring the median difference in particles between different stakeholders before and after inference, then we can report initial and final divergence in stakeholder opinion.
- *y metrics:* Stakeholders writing i* and feature models like Figure 1 and Figure 2 often explore multiple goals, where the y goals might be contradictory. We need some trade-off predicate that knows how to report improvements on multiple dimensions. We will use the Equation2 calculation from Table 3. As mentioned above, this calculation is preferred to standard "boolean domination" (one thing is better than another if it is no worse on any criteria and better on at least one criterion), since it is known that boolean domination can fail for three or more goals [97,109].
- *Acceptance metrics:* y metrics only comment on terms in a model. Above and beyond that, we need our post-surveys to capture other concerns from stakeholders about the acceptability of our final models.
- *Nonhuman-y metrics:* To calibrate and baseline the *ADVICE* y-metrics, we collect output y values seen in the AI tool (e.g. FLASH, HYPEROPT and OPTUNA [8,12,81]) make all the decisions themselves, with no human involvement.

5 Research Questions and Plan

Our research questions will be explored according to the research plan in Table 5. Note the last line of that table: *broadening participation in computing*. Just to explain that line, one of the benefits of NSF funding is the opportunity to work on broadening participating in computing (BPC). Our BPC plans are discussed in §6.3. As seen in our timetable, BPC will be an ongoing task through-out the work

Recall that this proposal will create the following systems:

- *ADVICE*-0: no team, no cloud (already built);
- *ADVICE*-1: adds support for team of stakeholders;
- *ADVICE*-2: adds use of cloud knowledge;
- *ADVICE*-3: adds verification tools;
- *ADVICE*-4: explores bias mitigation.

These different systems explore the following research questions (in the years, shown below). As said above, an important aspect of this research plan is its *survivability*. Given the multiple goals of *ADVICE*-1 .. *ADVICE*-4, if we fail on any one goal, we can still continue to succeed on the others.

RQ0: (Year 1,2,3) *Do people and* ADVICE *find different solutions for completing design tasks?*
This is one of our baseline sanity checks. Based on our current experience with *ADVICE*-0 [67, 68], it would seem that RQ1=yes. That said, this needs to be checked on each new model explored by this system.

RQ1: (Year 1,2,3) *Do humans using* ADVICE *find better results than (a) humans working fully manually* **AND** *(b) running AI tools without human-in-the-loop?*
This is our other baseline sanity check. If RQ1=no then that would mean that AI+human does worse than applying either, separately. Current results [67, 68] with *ADVICE*-0 suggest RQ1=yes but, as above, this needs to be checked on each new model explored by this system.
Relevant metrics: y-metrics, the acceptance metrics, and the nonhuman y-metrics.

RQ2: (Year 2) *For* ADVICE*-1, how to weight PSO's cognitive versus social reasoning?*
Recall that PSO determines the new velocity of a particle from the sum of a personnel's cognitive weight ϕ_p and the group social weight ϕ_g. The useful weights for this work would have to be determined by experimentation (perhaps learned via a self-adaptive scheme [37] or some configuration hyperparameter optimization [80]) but, initially, we take advice from Pedersen [87] who recommends $\phi_p + \phi_g \approx 4$ (after all distance weights are normalized 0..1).
Relevant metrics: The right $\phi_p + \phi_g$ optimizes for both the y-metrics and the acceptance metrics.

		Y1	Y2	Y3
RQ0	Do people and *ADVICE*- find different solutions for completing design tasks?	x	x	x
RQ1	Do humans using *ADVICE*- perform better than (a) humans, fully manually (b) AND AI tools without humans?	x	x	x
RQ2	For *ADVICE*-1, how to weigh PSO's cognitive versus social reasoning?		x	
RQ3	Is *ADVICE*-1,'s team support "useful"?		x	
RQ4	Is there a max "team size" threshold above which *ADVICE*-1 would not be recommended?		x	
RQ5	Are different kinds of cloud knowledge most useful for *ADVICE*-2?		x	
RQ6	Do different categories of stakeholders need different support for their different cognitive styles?		x	
RQ7	For *ADVICE*-3 Can meta-patterns of advice support verification?			x
RQ8	Are *ADVICE*'s models biased against specific social groups?			x
RQ9	How does *ADVICE*-4 effect model bias (if at all)?			x
RQ10	With *ADVICE*-4, does mitigating bias mean damaging performance?			x
RQ11	Can *ADVICE* support test case generation?			x
Also	Work on BPC (broadening participation in computing).	x	x	x

Table 5: Research Plan.

RQ3: (Year 2) *Is* ADVICE*-1's team support "useful"?*
Here, by "useful" we mean solutions with low divergence; good y-value scores; and which are described as acceptable by stakeholders. This RQ checks if PSO can negotiate shared solutions.
Relevant metrics: y-metrics, acceptance metrics, task completion time, and initial and final convergence.

RQ4: (Year 2) *Is there a max "team size" threshold above which* ADVICE*-1 would* not *be recommended?*
This RQ will explore what is effective team size. What we suspect here is that our methods will work up to some threshold "max team size" and then there will be some breakdown. Ideally, we will see a gradual (no sudden) breakdown– but that needs to be confirmed via experimentation.
Relevant metrics: (size of team) versus: (y-metrics, acceptance metrics, task completion time).

RQ5: (Year 2) *Are different kinds of cloud knowledge most useful for* ADVICE*-2?*
Table 1 lists a wide variety of cloud knowledge sources. In practice, only some (or none) of them might be accessible, or useful. To answer RQ5 we would try different cloud knowledge sources and, using the same metrics as RQ3, compare *ADVICE*2 results to *ADVICE*1.
Relevant metrics: Same as RQ3.

RQ6: (Year 2) *Do different categories of stakeholders need different support for their different cognitive styles?*
For this RQ, we pause in our implementation of *ADVICE*-1 to *ADVICE*-4 to check if our tools need different styles of interfaces. Prior results from co-PI Kuttal shows that different categories of stakeholders can have different decision-making styles [45] (for example, women participants perceived g constructive feedback from another woman while men participants perceived hurtful feedback from another man). If so, then this would have two consequences. Firstly, it might mean we should engineer different model browsing environments for different categories of stakeholders. Secondly, those different styles might suggest better algorithms for the internals of our search.
Relevant metrics: All the RQ2 metrics, divided according to our Stakeholder category metrics.

RQ7: (Year 3) *For* ADVICE*-3 Can meta-patterns of advice support verification?*
§3.3 discussed how advice "patterns" of (a) how often it is revised and (b) how that advice effects our ability to predict progress in the design process, and (c) other patterns.This RQ will check if we can find those patterns and if we can use them to select better advice, or mitigate for bad advice.
Relevant metrics: Same as RQ3.

RQ8: (Year 3) *Are* ADVICE*'s models biased againt specific social groups?*
Before we explore bias mitigation (in the next RQ), first we must check if our models exhibit bias.
Relevant metrics: There are at least two ways to detect bias in a model: (1) use the bias measures of §3.4 ; (2) check the post-surveys for any bias-related concerns (as reported by our participants).

RQ9: (Year 3) *How does* ADVICE*-4 effect model bias (if at all)?*
§3.4 discussed methods for model bias mitigation including personnel balancing, extra context, hyperparameter optimization, and sample balancing. (Aside: recall that "extra context" really means dialing up, or down, how much information we import from Table 1.). Here we would deploy teams with various degrees of diversity (as measured by our stakeholder category metrics). Those teams would be supported by different combinations of extra context, hyperparameter optimization and sample balancing. Note that this would be a very large set of case studies.
Relevant metrics: Same as RQ8.

RQ10: (Year 3) *With* ADVICE*-4, does mitigating bias mean damaging performance?*
Recall the concern raised above by Berk et al. [13] who said "It is impossible to achieve fairness and high performance simultaneously (except in trivial cases". At FSE'21, PI Menzies and his student Joymallya Chakraborty found that Berk et al. were needlessly pessimistic, e.g., sample balancing

to mitigate bias while preserving predictive performance. But that result was only for classification systems. In this RQ, we need to check if bias mitigation damages performance in multi-goal optimization. Note that if we see such damage, then it might be repairable (by extending the hyperparameter optimization methods of §3.4 to explore not just bias metrics, but y value performance metrics as well.

Relevant metrics: Same as RQ3, plus the bias metrics.

RQ11: (Year 3) *Can* ADVICE *support test case generation?*

(will explore if time permits). At its core, *ADVICE* explores a design looking for settings that most select for different behaviors. If we turn off user input and ran *ADVICE* N times with different random number seeds, then that would generate a set of vectors that separate the data into its most critical regions. This is clearly a way to generate tests that are spread across the input space of a model. In this RQ we would conduct a literature review to (a) document the current state-of-the-art (SOTA) in model-based test generation, and (b) discover what is current thinking on how to evaluate test coverage. We would then apply those coverage metrics to tests generated by *ADVICE* or SOTA.

Relevant metrics: Test coverage metrics, as documented by the literature review of this RQ.

6 Intellectual Merit and Broader Impact

6.1 Broader Impacts

Enabling humans to explore all the important potential behaviors of a software model is an open and important issue. Unfortunately, there are substantial examples where human oversight missed important software properties (see Table 2).

The project proposes advanced *collaborative discovery* software for students, faculty, and industry researchers in software engineering, artificial intelligence, and human-computer interaction, to realize a solution that can transform complex decision-making for stakeholders.

6.2 Intellectual Merit

The proposed research is original (none of the SAT problems in software engineering consider the team and cloud intelligence), transformative (potential to change decision making when designing software), and feasible (validated by user studies and preliminary designs). Specific aspects of the research that contribute to its intellectual merits are: (1) Identify specific cues related to social and cognition; modeling these to facilitate the decision-making process of stakeholders. (2) Tight data collection, design, evaluation, and refinement cycles for *ADVICE* models and they will incrementally evolve *ADVICE* functionality. (3) Diverse and focused empirical user studies will drive *ADVICE* design and refinement. (4) Overall research approach can be applied to SAT problems to help make decisions. The approaches involve identifying cues from individual humans and from logs, defining the inclusive model, verifying the results, create appropriate messages and interactions.

6.3 BPC Work: Broader Participation in Computer Science

This work was strongly motivated by co-PI Menzies and Kuttal's ongoing commitment to broader participation in computer science. This BPI work is strongly supported by the NC State's Computer Science department. Our department has a strong record of studying research issues related to gender bias [105], barriers faced by women [28], and methods for broadening participation [64] in the context of software engineering.

Co-PI Kuttal is very active in the GenderMag research, working towards reducing bias against women and other social groups in SE [5,45,53,65,91]) as well as involved in leading Society of Women Engineers (SWE), and NCWIT at the university and community level. PI Menzies is a member of his department's Broadening Participation Committee (BPC) that actively seeks to understand factors that make computer

science less attractive to underrepresented groups, then educate faculty, staff, and students on how different behaviors affect diversity, quality, and inclusiveness. Also, PI Menzies will continue his established tradition of graduating research students for historically under-represented groups.

As to this grant, whenever we can find factors discovered that block Broader Participation, those factors need to be documented and mitigated. For example, several of our research questions directly address issues of diversity and inclusion (see **RQ6, RQ8, RQ9, RQ10** in §5).

Other BPC activities supported by this work will be:

- Admission procedures for the NC State Department of Computing Science will be modeled by the tools created here, then reviewed by diverse groups from different social groups – from faculty, administration, and students at this department.
- The creation and extension of BPC-oriented lecture notes of the various NC State NSF-funded REUs (research experience for undergraduates) as well as graduate SE classes (taught by PI Menzies and co-PI Kuttal).
- Funds from this work will be used to support students attending the Grace Hopper conference, and the Richard Tapia Celebration of Diversity in Computing.

6.4 Dissemination of Knowledge

All the code developed as part of this work will be released as open-source software on GitHub, under an MIT license. Included in those packages will be the data used to certify the scripts as well as *RQn.sh* files containing executable scripts to reproduce (e.g.) RQ1.

As to papers, the PIs of this grant frequently published in top-ranked international scientific venues. Using those forums, the PIs will evangelize *ADVICE* in the software industry to increase productivity and empowerment. The results of the research and educational activities will be disseminated widely via publications and conference presentations, as well as software that will be used by researchers and educators.

Also, this work will generate much material (tools, scripts, data sets) that can be utilized by other research teams. For a decade, PI Menzies has lead-by-example in the open science community (the PROMISE project and the ROSE initiative) which takes care not only to package and distribute research code but also to publish papers and tutorials on that material.

7 Prior Results

PI Menzies is an IEEE Fellow and has earned over $13 million dollars in peer-reviewed competitive grants ($6.4M from NSF, and the rest from a variety of other government and industrial sources). Google Scholar lists him as a top-ten researcher in many research areas including knowledge acquisition and analytics. Serving as committee chair, he has graduated 18 Ph.D. and 32 master students (by research). He currently supervises 10 Ph.D. students at NC State. He has served as an associated editor on all the major SE journals and from 2021 will be EIC of the Automated Software Engineering journal.

Co-PI Kuttal has a well-established track record of creating tools that support programmer creativity [50] and exploratory programming behavior [42,43], debugging web-based distributed programming [55–57], problem-solving techniques [42, 43], gender-specific behavior [5, 45, 65], socio-technical skills [6, 52, 58, 94, 99, 104, 108, 111] and communication styles [50]. She is currently developing conversational agent for programmers [5, 41, 50, 54, 86, 91, 92], supporting information foraging by utilizing agents' collective foraging behavior [6, 99, 108], and studying developer's brain-to-brain interactions when problem-solving in same- and mixed- gender pairs.

We include below notes on some of the most recent NSF grants.

PI Menzies worked on (a) CCF-1302216, 2013-2017, $271,553; (b) "SHF: Medium: Collaborative: Transfer Learning in Software Engineering"; (c) The **intellectual merit** of that work was to define novel methods for sharing data, many of which were the precursor to the methods of this proposal. That work generated

the publications (d) [30, 39, 47–49, 77, 88] concerning prediction and planning methods. The **broader impact** of that work was to enable a new kind of open science– one where all data is routinely shared and is capable of building effective models no matter if it is obfuscated for security purposes. The methods of this project, while targeted at software engineering, could also be applied to any other data-intensive field. (e) Data from that work is now housed in the two publicly accessible repositories[7]. That work funded two Ph.D.s at NCSU. (f) N/A.

Another relevant research grant is (a) OAC-1826574, 2018-2018, $124,628; (b) "EAGER: Empirical Software Engineering for Computational Science"; (c) The **intellectual merit** of that work was to conduct initial explorations into novel methods for adapting SE methods to computational science. That work lead to the curious results that, in many ways, the computational scientists are better at managing their development cycle than many SE projects [107]. Whenever we found good enough data to compare the results seen in open source and computational science projects, we often find higher productivity values (and faster debugging) in computational science than in software engineering. (d) That work generated one journal paper (at TSE'21), one conference paper (at MSR'21) and another journal publication under review [63]. (e) Data from that work is now housed at the SEACRAFT publicly accessible repository [76]. That work funded one Ph.D. at NCSU. (f) N/A.

Co-PI Kuttal has received NSF CAREER and AFSoR YIP awards. Both projects are relevant to this project. Co-PI Kuttal is working on (a) IIS - 204620, 2021-2026, $ 520,522; (b) " HCC: CAREER: Designing an Interactive Partner to Support Pair Programming"; (c) The **intellectual merit** of that work was to create design guidelines for an anthropomorphic pair programming conversational agent [50, 54, 92], create initial set of 26 labels for programmer-agent conversations [86], establish feasibility of using machine learning models [91] and transformer-based language models for detecting programmer-agent conversations [86], and establish feasibility of using online videos of pair programming sessions for training a pair programming conversational agent [5, 41]. The **broader impact** of that work was to enable a new kind of open science– create an initial set of programmer-programmer and programmer-agent conversations. (e) Data from that work is publicly available[8]. That work funded one M.S. and one Ph.D., and two REUs at the University of Tulsa. Four other undergraduate students (2 women of color) worked on the research project. All students gained experience that crosses traditional boundaries of HCI, SE, and AI. (f) N/A.

Another relevant grant co-PI Kuttal is working on is (a) FA9550-21-1-0108, 2021-2024, $ 448,754$; (b) " Supporting Information Foraging by Utilizing Agents' Collective Foraging Behavior "; (c) The **intellectual merit** of that work developed generic model for Question & Answer website – Stack Overflow, foraging behavior of individual developers on GitHub and Stack Overflow [6]. It compared foraging patterns based on gender, specifically men and women. We are currently developing models for GitHub [99], Stack Overflow, and web to help foraging for these heterogeneous sources to facilitate the foraging of newcomers [108]. The **broader impact** of that work allowed students to gain experience that crosses traditional boundaries of HCI, SE, and AI. The project funded one Ph.D. student at the University of Tulsa. Eight other undergraduate students (2 women of color and 1 man of color) also worked on this research project. (f) N/A.

[7] **github.com/rshu/Adversarial-Evasion-Defense**

[8]

References

[1] "Amazon just showed us that 'unbiased' algorithms can be inadvertently racist," 2016. [Online]. Available: https://www.businessinsider.com/how-algorithms-can-be-racist-2016-4

[2] "Machine bias," *www.propublica.org*, May 2016. [Online]. Available: https://www.propublica.org/article/machine-bias-risk-assessments-in-criminal-sentencing

[3] "Study finds gender and skin-type bias in commercial artificial-intelligence systems," 2018. [Online]. Available: http://news.mit.edu/2018/study-finds-gender-skin-type-bias-artificial-intelligence-systems-0212

[4] "Software process line as an approach to support software process reuse: A systematic literature review," *Information and Software Technology*, vol. 116, p. 106175, 2019.

[5] J. H. A. McAuliffe and S. K. Kuttal, "Evaluating gender effects in pair programming conversations," in *VL/HCC*, 2022.

[6] B. M. A. Sedhain, S. S. Ragavan and S. K. Kuttal, "Estimating foraging values and costs in stack overflow," in *VL/HCC*, 2022.

[7] A. Agrawal, W. Fu, and T. Menzies, "What is wrong with topic modeling? and how to fix it using search-based software engineering," *Information and Software Technology*, vol. 98, pp. 74–88, jun 2018. [Online]. Available: https://doi.org/10.1016%2Fj.infsof.2018.02.005

[8] T. Akiba, S. Sano, T. Yanase, T. Ohta, and M. Koyama, "Optuna: A next-generation hyperparameter optimization framework," in *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*, 2019, pp. 2623–2631.

[9] A. A. Araújo, M. Paixao, I. Yeltsin, A. Dantas, and J. Souza, "An architecture based on interactive optimization and machine learning applied to the next release problem," *Automated Software Engineering*, vol. 24, no. 3, pp. 623–671, 2017.

[10] E. R. Batot, S. Gérard, and J. Cabot, "A survey-driven feature model for software traceability approaches," in *International Conference on Fundamental Approaches to Software Engineering*. Springer, Cham, 2022, pp. 23–48.

[11] N. Bencomo, J. L. Guo, R. Harrison, H.-M. Heyn, and T. Menzies, "The secret to better ai and better software (is requirements engineering)," *IEEE Software*, vol. 39, no. 1, pp. 105–110, 2022.

[12] J. Bergstra, B. Komer, C. Eliasmith, D. Yamins, and D. D. Cox, "Hyperopt: a python library for model selection and hyperparameter optimization," *Computational Science & Discovery*, vol. 8, no. 1, p. 014008, 2015.

[13] R. Berk, H. Heidari, S. Jabbari, M. Kearns, and A. Roth, "Fairness in criminal justice risk assessments: The state of the art," *Sociological Methods Research*, 03 2017.

[14] M. Bhushan, A. Negi, P. Samant, S. Goel, and A. Kumar, "A classification and systematic review of product line feature model defects," *Software Quality Journal*, vol. 28, no. 4, p. 1507–1550, dec 2020. [Online]. Available: https://doi.org/10.1007/s11219-020-09522-1

[15] A. Biere, "Picosat essentials," *Journal on Satisfiability, Boolean Modeling and Computation*, vol. 4, no. 2-4, pp. 75–97, 2008.

[16] V. Braun and V. Clarke, "Using thematic analysis in psychology," *Qualitative Research in Psychology*, vol. 3, no. 2, pp. 77–101, 2006. [Online]. Available: https://www.tandfonline.com/doi/abs/10.1191/1478088706qp063oa

[17] P. Bresciani, A. Perini, P. Giorgini, F. Giunchiglia, and J. Mylopoulos, "Tropos: An agent-oriented software development methodology," *Autonomous Agents and Multi-Agent Systems*, vol. 8, no. 3, pp. 203–236, 2004.

[18] J. Burstein, C. Doran, and T. Solorio, Eds., *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*. Association for Computational Linguistics, 2019. [Online]. Available: https://aclanthology.org/volumes/N19-1/

[19] A. Caliskan, J. J. Bryson, and A. Narayanan, "Semantics derived automatically from language corpora contain human-like biases," *Science*, vol. 356, no. 6334, pp. 183–186, 2017. [Online]. Available: https://science.sciencemag.org/content/356/6334/183

[20] M. Camilli and B. Russo, "Modeling performance of microservices systems with growth theory," *Empirical Software Engineering*, vol. 27, no. 2, pp. 1–44, 2022.

[21] J. Chakraborty, "Fairway," 6 2020. [Online]. Available: https://figshare.com/articles/software/Fairway/12521408

[22] J. Chakraborty, S. Majumder, and T. Menzies, "Bias in machine learning software: Why? how? what to do?" in *FSE'21*, 05 2021.

[23] J. Chakraborty, S. Majumder, Z. Yu, and T. Menzies, "Fairway: A way to build fair ml software," in *Proceedings of the 28th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, ser. ESEC/FSE 2020. New York, NY, USA: Association for Computing Machinery, 2020, p. 654–665. [Online]. Available: https://doi.org/10.1145/3368089.3409697

[24] O. Chapelle, B. Schölkopf, and A. Zien, Eds., *Semi-Supervised Learning*. The MIT Press, 2006. [Online]. Available: http://dblp.uni-trier.de/db/books/collections/CSZ2006.html

[25] A. Chatterjee, M. Guizani, C. Stevens, J. Emard, M. E. May, M. Burnett, and I. Ahmed, "Aid: An automated detector for gender-inclusivity bugs in oss project pages," in *2021 IEEE/ACM 43rd International Conference on Software Engineering (ICSE)*, 2021, pp. 1423–1435.

[26] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: Nsga-ii," *IEEE transactions on evolutionary computation*, vol. 6, no. 2, pp. 182–197, 2002.

[27] M. S. Feather and T. Menzies, "Converging on the optimal attainment of requirements," in *10th Anniversary IEEE Joint International Conference on Requirements Engineering (RE 2002), 9-13 September 2002, Essen, Germany*. IEEE Computer Society, 2002, pp. 263–272. [Online]. Available: https://doi.org/10.1109/ICRE.2002.1048537

[28] D. Ford, J. Smith, P. J. Guo, and C. Parnin, "Paradise unplugged: Identifying barriers for female participation on stack overflow," in *Proceedings of the 2016 24th ACM SIGSOFT International Symposium on Foundations of Software Engineering*, 2016, pp. 846–857.

[29] M. Fu and C. Tantithamthavorn, "Gpt2sp: A transformer-based agile story point estimation approach," *IEEE Transactions on Software Engineering*, no. 01, pp. 1–1, mar 5555.

[30] W. Fu, T. Menzies, and X. Shen, "Tuning for software analytics: Is it really necessary?" *Information and Software Technology*, vol. 76, pp. 135–146, 2016.

[31] T. Gebru, ""race and gender."

[32] B. G. Glaser and A. L. Strauss, *The Discovery of Grounded Theory: Strategies for Qualitative Research*. New York, NY: Aldine de Gruyter, 1967.

[33] B. Green, "The flaws of policies requiring human oversight of government algorithms," *Computer Law & Security Review*, vol. 45, p. 105681, 2022.

[34] M. Guizani, L. Letaw, M. Burnett, and A. Sarma, "Gender inclusivity as a quality requirement: Practices and pitfalls," *IEEE Software*, vol. 37, no. 6, pp. 7–11, 2020.

[35] X. Han, Z. Zhang, N. Ding, Y. Gu, X. Liu, Y. Huo, J. Qiu, Y. Yao, A. Zhang, L. Zhang, W. Han, M. Huang, Q. Jin, Y. Lan, Y. Liu, Z. Liu, Z. Lu, X. Qiu, R. Song, J. Tang, J.-R. Wen, J. Yuan, W. X. Zhao, and J. Zhu, "Pre-trained models: Past, present and future," *AI Open*, vol. 2, pp. 225–250, 2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2666651021000231

[36] M. Harman, S. A. Mansouri, and Y. Zhang, "Search-based software engineering: Trends, techniques and applications," *ACM Comput. Surv.*, 2012.

[37] K. R. Harrison, A. P. Engelbrecht, and B. M. Ombuki-Berman, "Self-adaptive particle swarm optimization: a review and analysis of convergence," *Swarm Intelligence*, vol. 12, no. 3, pp. 187–226, 2018.

[38] S. G. Hart, "Nasa-task load index (nasa-tlx); 20 years later," in *Proceedings of the human factors and ergonomics society annual meeting*, vol. 50, no. 9. Sage publications Sage CA: Los Angeles, CA, 2006, pp. 904–908.

[39] Z. He, F. Peters, T. Menzies, and Y. Yang, "Learning from open-source projects: An empirical study on defect prediction," in *Empirical Software Engineering and Measurement, 2013 ACM/IEEE International Symposium on*. IEEE, 2013, pp. 45–54.

[40] J. Horkoff and E. Yu, "Interactive goal model analysis for early requirements engineering," *Requirements Engineering*, vol. 21, no. 1, pp. 29–61, 2016.

[41] J. A. J. Hart and S. K. Kuttal, "Feasibility of using youtube conversations for pair programming intent classification," in *VL/HCC*, 2022.

[42] W. Jernigan, A. Horvath, M. Lee, M. Burnett, C. Taylor, S. Kuttal, A. Peters, I. Kwan, F. Bahmani, and A. Ko, "A principled evaluation for a principled idea garden," 10 2015.

[43] W. Jernigan, A. Horvath, M. Lee, M. Burnett, T. Cuilty, S. Kuttal, A. Peters, I. Kwan, F. Bahmani, A. Ko, and C. Mendez, "General principles for a generalized idea garden," *Journal of Visual Languages & Computing*, 05 2017.

[44] K. C. Kang, S. G. Cohen, J. A. Hess, W. E. Novak, and A. S. Peterson, "Feature-oriented domain analysis (foda) feasibility study," Carnegie-Mellon Univ Pittsburgh Pa Software Engineering Inst, Tech. Rep., 1990.

[45] S. Kaur Kuttal, K. Gerstner, and A. Bejarano, "Remote pair programming in online cs education: Investigating through a gender lens," in *VL/HCC*, 2019, pp. 75–85.

[46] J. Kennedy and R. C. Eberhart, "Particle swarm optimization," in *Proceedings of the IEEE International Conference on Neural Networks*, 1995, pp. 1942–1948.

[47] R. Krishna, V. Nair, P. Jamshidi, and T. Menzies, "Whence to learn? transferring knowledge in configurable systems using beetle," *TSE*, 2020.

[48] R. Krishna and T. Menzies, "Learning effective changes for software projects," *arXiv preprint arXiv:1708.05442. Under review; IEEE TSE*, 2017.

[49] ——, "Bellwethers: A baseline method for transfer learning," *IEEE Transactions on Software Engineering*, 2018.

[50] S. K. Kuttal, J. Myers, S. Gurka, D. Magar, D. Piorkowski, and R. Bellamy, "Towards designing conversational agents for pair programming: Accounting for creativity strategies and conversational styles," in *VL/HCC*, 2020, pp. 1–11.

[51] S. K. Kuttal, Y. Bai, E. Scott, and R. Sharma, "Tug of perspectives: Mobile app users vs developers," 2020.

[52] S. K. Kuttal, X. Chen, Z. Wang, S. Balali, and A. Sarma, "Visual resume: Exploring developers' online contributions for hiring," *Inf. Softw. Technol.*, vol. 138, p. 106633, 2021. [Online]. Available: https://doi.org/10.1016/j.infsof.2021.106633

[53] S. K. Kuttal, S. Y. Kim, C. Martos, and A. Bejarano, "How end-user programmers forage in online repositories? an information foraging perspective," *J. Comput. Lang.*, vol. 62, p. 101010, 2021. [Online]. Available: https://doi.org/10.1016/j.cola.2020.101010

[54] S. K. Kuttal, B. Ong, K. Kwasny, and P. Robe, "Trade-offs for substituting a human with an agent in a pair programming context: The good, the bad, and the ugly," in *CHI*, 2021.

[55] S. K. Kuttal, A. Sarma, M. Burnett, G. Rothermel, I. Koeppe, and B. Shepherd, "How end-user programmers debug visual web-based programs: An information foraging theory perspective," *J. Comput. Lang.*, vol. 53, pp. 22–37, 2019. [Online]. Available: https://doi.org/10.1016/j.cola.2019.04.003

[56] S. K. Kuttal, A. Sarma, and G. Rothermel, "Debugging support for end user mashup programming," in *2013 ACM SIGCHI Conference on Human Factors in Computing Systems, CHI '13, Paris, France, April 27 - May 2, 2013*, 2013, pp. 1609–1618. [Online]. Available: https://doi.org/10.1145/2470654.2466213

[57] ——, "Predator behavior in the wild web world of bugs: An information foraging theory perspective," in *2013 IEEE Symposium on Visual Languages and Human Centric Computing, San Jose, CA, USA, September 15-19, 2013*, 2013, pp. 59–66. [Online]. Available: https://doi.org/10.1109/VLHCC.2013.6645244

[58] S. K. Kuttal, A. Sarma, G. Rothermel, and Z. Wang, "What happened to my application? helping end users comprehend evolution through variation management," *Inf. Softw. Technol.*, vol. 103, pp. 55–74, 2018. [Online]. Available: https://doi.org/10.1016/j.infsof.2018.06.008

[59] S. K. Kuttal, A. Sarma, A. Swearngin, and G. Rothermel, "Versioning for mashups - an exploratory study," in *End-User Development - Third International Symposium, IS-EUD 2011, Torre Canne (BR), Italy, June 7-10, 2011. Proceedings*, 2011, pp. 25–41. [Online]. Available: https://doi.org/10.1007/978-3-642-21530-8_4

[60] S. Leavy, "Gender bias in artificial intelligence: The need for diversity and gender theory in machine learning," in *2018 IEEE/ACM 1st International Workshop on Gender Equality in Software Engineering (GE)*, 2018, pp. 14–16.

[61] C. Lewis, *Using the "thinking Aloud" Method in Cognitive Interface Design*, ser. Research report. IBM T.J. Watson Research Center. [Online]. Available: https://books.google.com/books?id=F5AKHQAACAAJ

[62] B. Lin, F. Zampetti, G. Bavota, M. Di Penta, M. Lanza, and R. Oliveto, "Sentiment analysis for software engineering: How far can we go?" in *2018 IEEE/ACM 40th International Conference on Software Engineering (ICSE)*, 2018, pp. 94–104.

[63] X. Ling, R. Agrawal, and T. Menzies, "How different is test case prioritization for open and closed source projects," *IEEE Transactions on Software Engineering*, mar 2021.

[64] F. Liu, D. Ford, C. Parnin, and L. Dabbish, "Selfies as social movements: Influences on participation and perceived impact on stereotypes," *Proc. ACM Hum.-Comput. Interact.*, no. CSCW, 2017. [Online]. Available: https://doi.org/10.1145/3134707

[65] C. Lott, A. McAuliffe, and S. K. Kuttal, "Remote pair collaborations of cs students: Leaving women behind?" in *VL/HCC*, 2021.

[66] ——, "Remote pair collaborations of cs students: Leaving women behind?" in *2021 IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC)*, 2021, pp. 1–11.

[67] A. Lustosa and T. Menzies, "Look before you leap: Predicting open-source project health (using landscape analysis)," in *submitted to ASE'22*.

[68] A. Lustosa, J. Patel, V. S. T. Malapati, and T. Menzies, "Sneak: Faster interactive search-based se," 2021. [Online]. Available: https://arxiv.org/abs/2110.02922

[69] D. S. Mai, L. T. Ngo, L. H. Trinh, and H. Hagras, "A hybrid interval type-2 semi-supervised possibilistic fuzzy c-means clustering and particle swarm optimization for satellite image analysis," *Information Sciences*, vol. 548, pp. 398–422, 2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0020025520309920

[70] S. Majumder, J. Chakraborty, G. R. Bai, K. T. Stolee, and T. Menzies, "Fair enough: Searching for sufficient measures of fairness," 2021. [Online]. Available: https://arxiv.org/abs/2110.13029

[71] C. Martos, S. Y. Kim, and S. K. Kuttal, "Reuse of variants in online repositories: Foraging for the fittest," in *2016 IEEE Symposium on Visual Languages and Human-Centric Computing, VL/HCC 2016, Cambridge, United Kingdom, September 4-8, 2016*, 2016, pp. 124–128. [Online]. Available: https://doi.org/10.1109/VLHCC.2016.7739674

[72] G. Mathew, T. Menzies, N. A. Ernst, and J. Klein, "Shorter reasoning about larger requirements models," in *25th IEEE International Requirements Engineering Conference*, 2017.

[73] ——, ""short"er reasoning about larger requirements models," in *RE'17*, 2017. [Online]. Available: https://arxiv.org/abs/1702.05568

[74] R. Medeiros, "Unburdening onboarding in software product lines," in *2021 IEEE/ACM 43rd International Conference on Software Engineering: Companion Proceedings (ICSE-Companion)*, 2021, pp. 260–262.

[75] M. Mendonca, M. Branco, and D. Cowan, "Splot: software product lines online tools," in *Proceedings of the 24th ACM SIGPLAN conference companion on Object oriented programming systems languages and applications*, 2009, pp. 761–762.

[76] T. Menzies, R. Krishna, and D. Pryor, "The seacraft repository of empirical software engineering data," *Retrieved March*, 2017.

[77] T. Menzies, W. Nichols, F. Shull, and L. Layman, "Are delayed issues harder to resolve? revisiting cost-to-fix of defects throughout the lifecycle," *Empirical Software Engineering*, vol. 22, no. 4, pp. 1903–1935, 2017.

[78] T. Menzies, D. Owen, and J. Richardson, "The strangest thing about software," *Computer*, vol. 40, no. 1, pp. 54–60, 2007.

[79] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," *CoRR*, vol. abs/1301.3781, 2013. [Online]. Available: http://dblp.uni-trier.de/db/journals/corr/corr1301.html#abs-1301-3781

[80] V. Nair, Z. Yu, T. Menzies, N. Siegmund, and S. Apel, "Finding faster configurations using flash," *TSE*, pp. 1–1, 2018.

[81] V. Nair, R. Krishna, T. Menzies, and P. Jamshidi, "Transfer learning with bellwethers to find good configurations," *CoRR*, 2018.

[82] J. Nielsen and R. Molich, "Heuristic evaluation of user interfaces," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, ser. CHI '90. New York, NY, USA: Association for Computing Machinery, 1990, p. 249–256. [Online]. Available: https://doi.org/10.1145/97243.97281

[83] S. U. Noble, *Algorithms of oppression*. New York University Press, 2018.

[84] I. Northpointe, "Practitioner's guide to compas core." 2015.

[85] N. Novielli, F. Calefato, and F. Lanubile, "The challenges of sentiment detection in the social programmer ecosystem," in *Proceedings of the 7th International Workshop on Social Software Engineering*, 2015, pp. 33–40.

[86] J. A. P. Robe, S. K. Kuttal and J. Har, "Pair programming conversations with agents vs. developers: Challenges & opportunities for se community," in *The ACM Joint European Software Engineering Conference, and Symposium on the Foundations of Software Engineering*, 2022.

[87] M. E. H. Pedersen, "Good parameters for particle swarm optimization," *Hvass Lab., Copenhagen, Denmark, Tech. Rep. HL1001*, pp. 1551–3203, 2010.

[88] F. Peters, T. Menzies, and L. Layman, "Lace2: Better privacy-preserving data sharing for cross project defect prediction," in *2015 IEEE/ACM 37th IEEE International Conference on Software Engineering*, vol. 1. IEEE, 2015, pp. 801–811.

[89] S. S. Ragavan, B. Pandya, D. Piorkowski, C. Hill, S. K. Kuttal, A. Sarma, and M. M. Burnett, "PFIS-V: modeling foraging behavior in the presence of variants," in *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems, Denver, CO, USA, May 06-11, 2017*, 2017, pp. 6232–6244. [Online]. Available: https://doi.org/10.1145/3025453.3025818

[90] A. J. Ramirez, A. C. Jensen, and B. H. C. Cheng, "A taxonomy of uncertainty for dynamically adaptive systems," in *2012 7th International Symposium on Software Engineering for Adaptive and Self-Managing Systems (SEAMS)*. IEEE, jun 2012, pp. 99–108. [Online]. Available: http://ieeexplore.ieee.org/document/6224396/

[91] P. Robe, S. Kaur Kuttal, Y. Zhang, and R. Bellamy, "Can machine learning facilitate remote pair programming? challenges, insights & implications," in *VL/HCC*, 2020, pp. 1–11.

[92] P. Robe and S. K. Kuttal, *Designing PairBuddy – Conversational Agent for Pair Programming*, 2022, vol. 29, p. 1–13.

[93] C. Rudin, "Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead," *Nature Machine Intelligence*, vol. 1, no. 5, pp. 206–215, 2019. [Online]. Available: https://doi.org/10.1038/s42256-019-0048-x

[94] A. Sarma, X. Chen, S. Kuttal, L. Dabbish, and Z. Wang, "Hiring in the global stage: Profiles of online contributions," in *ICGSE*, 2016, pp. 1–10.

[95] P. Sawyer, N. Bencomo, J. Whittle, E. Letier, and A. Finkelstein, "Requirements-aware systems: A research agenda for re for self-adaptive systems," in *2010 18th IEEE International Requirements Engineering Conference*, 2010, pp. 95–103.

[96] A. S. Sayyad, J. Ingram, T. Menzies, and H. Ammar, "Scalable product line configuration: A straw to break the camel's back," in *2013 28th IEEE/ACM International Conference on Automated Software Engineering (ASE)*, 2013, pp. 465–474.

[97] A. S. Sayyad, T. Menzies, and H. Ammar, "On the value of user preferences in search-based software engineering: A case study in software product lines," in *Proceedings of the 2013 International Conference on Software Engineering*, ser. ICSE '13. Piscataway, NJ, USA: IEEE Press, 2013, pp. 492–501. [Online]. Available: http://dl.acm.org/citation.cfm?id=2486788.2486853

[98] C. Seaman, "Qualitative methods in empirical studies of software engineering," *IEEE Transactions on Software Engineering*, vol. 25, no. 4, pp. 557–572, 1999.

[99] A. Sedhain and S. K. Kuttal, "Information seeking behavior for bugs on github: An information foraging perspective," in *VL/HCC*, 2022.

[100] B. Settles, "Active learning literature survey," 2009.

[101] B. Shneiderman, "Designing computer system messages," *Commun. ACM*, vol. 25, no. 9, p. 610–611, sep 1982. [Online]. Available: https://doi.org/10.1145/358628.358639

[102] H. A. Simon, "Rational choice and the structure of the environment," *Psychological Review*, vol. 63, no. 2, pp. 129–138, 1956.

[103] A. Strauss and J. Corbin, *Basics of qualitative research*. Sage publications, 1990.

[104] M. M. Sun, A. Ghosh, R. Sharma, and S. K. Kuttal, "Birds of a feather flock together? A study of developers' flocking and migration behavior in github and stack overflow," *CoRR*, vol. abs/1810.13062, 2018. [Online]. Available: http://arxiv.org/abs/1810.13062

[105] J. Terrell, A. Kofink, J. Middleton, C. Rainear, E. Murphy-Hill, C. Parnin, and J. Stallrich, "Gender differences and bias in open source: Pull request acceptance of women versus men," *PeerJ Computer Science*, 2017.

[106] H. Thaller, L. Linsbauer, and A. Egyed, "Feature maps: A comprehensible software representation for design pattern detection," *Saner'19*, vol. abs/1812.09873, 2019. [Online]. Available: http://arxiv.org/abs/1812.09873

[107] H. Tu, R. Agrawal, and T. Menzies, "The changing nature of computational science software," *arXiv preprint arXiv:2003.05922*, 2020.

[108] G. B. V. Diwanji, A. Sedhain and S. K. Kuttal, "Developers' foraging behavior on stack overflow," in *VL/HCC*, 2022.

[109] T. Wagner, N. Beume, and B. Naujoks, "Pareto-, aggregation-, and indicator-based methods in many-objective optimization," in *Proceedings of the 4th International Conference on Evolutionary Multi-criterion Optimization*, ser. EMO'07. Berlin, Heidelberg: Springer-Verlag, 2007, pp. 742–756. [Online]. Available: http://dl.acm.org/citation.cfm?id=1762545.1762608

[110] Z. Yu, C. Theisen, L. Williams, and T. Menzies, "Improving vulnerability inspection efficiency using active learning," *IEEE Transactions on Software Engineering*, vol. 47, no. 11, pp. 1–1, 2021.

[111] C. Zhou, S. K. Kuttal, and I. Ahmed, "What makes a good developer? an empirical study of developers' technical and social competencies," in *2018 IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC)*, 2018, pp. 319–321.

[112] E. Zitzler, M. Laumanns, and L. Thiele, "Spea2: Improving the strength pareto evolutionary algorithm for multiobjective optimization," in *Evolutionary Methods for Design, Optimisation, and Control*. CIMNE, Barcelona, Spain, 2002, pp. 95–100.

Facilities, Equipment, and Other Resources

Offices:

The project PIs has and offices in their CS Department. This department has adequate space to house all research assistants working on this project. All offices are wired for high-speed network access.

The PIs' departments at NC State provide the space and basic networking services to carry out the experiments, secretarial and administrative support as well as general-purpose office equipment (*e.g.*, fax, photocopiers, etc.).

Lab Space

The PIs have their own lab space at NC State. PI Menzies' RAISE lab (Real-World AI and SE) is a newly renovated space containing over 1,500 ft^2 of research space and 15 cubicles, a meeting space, printer, and wide screen projector.

Compute Facilities

Part of *ADVICE* will involve comparatively assessing different technologies. For that process, it will be useful to have some large-scale compute facility.

At NCSU, students working on this grant will have access to a 108-node compute cluster named ARC with 2,000 cores (AMD Mangy-Cours), Infiniband QDR interconnect, per node power monitoring, GPUs and SSDs and parallel file system support, which was funded by an NSF CRI that he is the main PI of together with 5 co-PIs. The ARC facility is providing local and remote researchers with administrator/root privileges for Computer Science experiments at a medium scale. This allows any of the software layers, including the operating system and Infiniband switch network routing tables, to be modified for experimental purposes, e.g., to experiment with different network topologies. For large-scale demonstrations, other facilities will be utilized (the HPC discussed below.

Additionally, NC State University provides a High-Performance Computing (HPC) facility as a part of the initiative to provide state-of-the-art support for research and academic computing. HPC system (called henry2) provides NC State students and faculty with entry and medium-level high-performance research and education computing facilities, consulting support and scientific workflow support. The HPC ecosystem consists of 1233 dual Xeon compute nodes in the henry2 cluster. Each node has two Xeon processors (mix of dual-, quad-, six-, eight-, ten-core, twelve-core) and 2 to 6 GigaBytes of memory per core. The total number of cores increases as more cores are purchased and now exceeds 10000. The nodes all have 64-bit processors. All HPC projects have the capability to run jobs using up to 128 processor cores for up to 48 hours and smaller jobs up to a week.

Data Management Plan

The PIs will take responsibility and provide oversight for the management of data associated with the proposed research. The students associated with this project will be briefed on and required to adhere to the data management plan. In addition, PIs and students associated with this project will receive appropriate IRB training and certification that includes information and best practices related to the ethical management of human subjects' data. This will allow the data generated from the project to be shared across institutions to the extent allowed by the IRB board at NCSU in accordance with NSF policies.

10.1 Data generated and access

As per our universities' IRB policies, the data throughout the research will be collected only from participants who agree. All materials will be made anonymous and stored locally under password protection, so as to be available only to the research team and managed under the restrictions imposed by the IRB approval obtained by the PIs' institution. In particular, data identifying participants will be erased completely as soon as anonymization is complete. Additional metadata will include coding schemes and memos, experiment protocols, design recommendations and derivative design ideas, and scenarios of our proposed tool (ADVICE). However, aggregations (descriptive and inferential statistics and selected, anonymized short excerpts of quotes, logs, or behavior sequences) will be disseminated through scholarly publication, as will details of the study designs, methodologies, and procedures used to collect these data.

10.1.1 Data from Labs and Case Studies

Data gathered in the controlled lab studies. This data will include qualitative (observational field notes, recorded participants' verbalizations and behaviors) and quantitative data (questionnaires, log data, digital audio/video recordings of studies with human subjects, screen captures of user experiment sessions, transcriptions of interview results, and programs) representing the performance, outcomes, and attitudes of each participant.

Data gathered from the case studies. The data will be collected from the classes with students' consent and from professionals using MTurk. The videos of participants will be transcribed and labeled with unique identifiers to anonymize them. The details can be found in Section 4.

Formative dialogue templates. The conversations collected from classes and Mturk will also be analyzed to identify decisions. The decision templates created from human data will be also made available on Project website. These templates won't contain any participant information.

10.1.2 ADVICE Usage Data

Computer programs written by the participants, including programs written in the programming languages supported by the tool. Unless their authors explicitly share these programs with the online community, these programs will be available only to the participants and the research team and managed under the restrictions imposed by the IRB approval obtained by the PI's institutions. The only programs that will be disseminated through scholarly publication will be ones shared by participants.

Participant's assessment results. This data will be available only to the participants and the research teams and managed under the restrictions imposed by the IRB approval obtained by the PI's institutions but will be disseminated in the aggregate through scholarly publication. Participants will have the option of publicly displaying their progress in the online community.

10.2 Dissemination and sharing of results

Key excerpts of anonymized primary data will be routinely shared in the context of publications of this research. Final coding schemes, design recommendations, and design prototypes that serve to synthesize across and characterize our holistic understanding of the primary data will be shared as well, in the context of publications of this work. All publications resulting from this research will include relevant methodological information necessary to understand how the data was collected and analyzed. In the event that key data has not been published in formal venues within 3 years of the termination of

this fellowship, all unpublished manuscripts will be made publically available as tech notes or working papers. Human subject's data that include privileged and confidential information and raw data will be kept secure in accordance to approved IRB protocols and best practices for protection of human subjects. Any human subject's data that is publicly released or shared will be suitably anonymized and aggregated to maintain confidentiality and protect the privacy of subjects.

We will promptly prepare and submit for publication, with authorship that accurately reflects the contributions of those involved, all significant findings from work performed in this project. To promote widespread use and dissemination of our research results, our first choices for publishing this research will be at venues that allow authors to make digital copies of their publications freely available via their personal websites (e.g., ACM conferences). To the extent that this is possible, then, all publications (including tech notes and working papers) resulting from this research will be made publically available on the project website as pdf files.

10.3 Software

A dedicated website will be created for the project and the entire project related annotated data, ADVICE software access, publications, and relevant documents will made accessible through the website. ADVICE will use standard software development practices and will be made open source under the GPL license. The ADVICE prototype will be distributed, when allowable by PIs' tech transfer office, with source code, and through project web site for at least three years, so that it is available to and modifiable by anyone in the world. The ADVICE tool source may also be hosted on an open source project site such as GitHub, which will allow our team to coordinate our development efforts as well as allow the general public to download the prototype as they see fit.

This research will also produce public source code; for example structured data sets; predictive models; predictive model's intermediate tuning cache; records of results of applying the code to the data; and work-processing files (the reports of our results). All data will be stored in different formats, appropriate for the type of data being stored. For example, appropriate data formats for papers include word files, latex files, and PDF files. As another example, appropriate data formats for repository data include Attribute Relation File Format (.arff) or Comma-Separated Values (.csv). Similarly Hierarchical Data Format (.h5) or Pickle(.pkl) for storing generated models. As to everything else, where possible, open source data and software will be shared publicly through a repository (with an exception for qualitative survey results which may be presented as a survey summary after applying de-identification of the responses from individuals).

Repository data will be freely available to the world and licensed under Creative Commons Attribution 4.0. International license (https://creativecommons.org/ licenses/by/4.0/). This data will be made freely available in our repository, accessible through the internet. All published and open source data may be re-used, re-distributed, and derived as long as the original data is not misrepresented, and the materials do not violate the acceptable use of the IP holders. Others may change or alter the open source data for personal purposes provided that these changes are made clear in any publications, re- distributions, or derivations.

Repository data will be hosted on the Zenodo repository (hosted at the Large Hadron Collider in Switzerland). Papers will be published at major conferences and journals in SE data analytics. Project results will be archived at the University on its web server, managed and supported by the college's IT team. Individual human-subjects data will be archived on a password-protected system or in locked cabinets accessible to only the PI and the research team. Raw data will be archived for a limited period of time of no more than 3 years from the completion of the project as required by approved human-subjects protocols; after this period of time has expired, the data will be destroyed to protect the privacy of individuals involved.

Project Personnel

1. Tim Menzies, North Carolina State University; PI
2. Sandeep Kuttal, North Carolina State University; co-PI