WASP Software Engineering: Reflective Essay

Alexander Gower (19930307–9132)

VT2023 (*Submitted 31 August 2023*)

Contents

---

1 Introduction

My research is aimed at automating scientific discovery in systems biology. The motivation for this problem comes from a need for a greater rate of discovery within biology to meet the challenges human society faces in the twenty-first century. Climate change, food security, global health and energy security are all pressing global issues. Research in biology can provide insight into these issues, and engineering solutions to help ameliorate them. However, if the rate of discovery in biology does increase then we risk insufficient progress to address these challenges.

The rate of discovery in biology is limited primarily by human capital. Discoveries require human effort in many aspects of the scientific process, for example: hypothesis generation; experiment design; experimental execution; sample extraction and preparation; data generation and analysis; and reporting.

By automating parts of the scientific process, one can achieve incremental improvements in efficiency. This is currently happening across the natural sciences. Much greater efficiencies can be achieved through "closed-loop discovery", whereby an autonomous system can execute cycles of scientific discovery independent of human intervention. Such AI researchers offer superhuman abilities in many areas, including reasoning, recall and precision.

Our lab focuses on automated scientific discovery in yeast systems biology, studying the yeast *Saccharomyces cerevisiae*. I develop models of *S. cerevisiae* that are amenable to automated improvement. Currently I am working on a model based in first-order predicate logic for the metabolism of the yeast cell. In tandem, I design algorithms to improve these models. The "fuel" for algorithmic improvement, and the discovery of new scientific knowledge, is experimental data. This data is either collated from previous studies, or generated using our yeast culturing platform at Chalmers.

2 How my research relates to course topics

I selected the following principles/ideas/concepts/techniques:

- Certification of systems for scientific discovery
- Versioning of data for AI-generated scientific models
- The need for domain-specific tools
- Devbots may be necessary for maintenance and development of domain knowledge.
- Project management
- Sustainability (of software products)

2.1 Validation and certification of software systems for scientific discovery

A problem that I have encountered in my research is developing good test cases for my scientific discovery systems. Good test cases in software engineering test the boundaries of the system and also have known results. High-level test cases would naturally be previously made scientific discoveries within yeast biology, e.g. "the gene ARG1 has the function of catalysing reactions in the 'L-arginine biosynthesis' pathway." Translating such high-level test cases to software requirements (a test-driven development approach) is not something I have tried, largely because when I began this project my software engineering skills and knowledge were not very advanced. But also translating these cases into lower-level tests, even retrospectively, is not a trivial task.

An additional challenge is certifying within the scientific community that the algorithmic approach made significant contribution to the scientific discovery, rather than being merely an aid. There are no established standards for scientific discovery algorithms—such certification takes place via the peer-review process that is used to certify scientific contributions from human researchers, e.g. when submitting to journals or conferences. This peer-review approach is likely not scalable to the discoveries made by AI scientists, and equally the ability to interrogate AI scientists to certify their method and conclusions is more limited than with humans.

In the future one could imagine a certification procedure for AI scientists, similar in spirit to the doctoral thesis and defence certification procedure for human researchers. How this type of certification would work for AI researchers is not a well-developed idea.

2.2 Versioning of scientific data for AI-generated scientific models

A related problem is that of specification of data used to drive scientific discovery algorithms. Experimental data is collected from a wide variety of conditions, labs, machines and protocols. Small variations can cause large differences in the resultant data, and therefore it is very important to know which version of a particular dataset was used to generate new knowledge. This is important for the reproducibility of science that is done by AI researchers.

2.3 Domain-specific tools

In his guest lecture Dr. Fabio Calefato talked about the increase in performance when using sentiment analysis tools that were tailored for software engineering, rather than "off-the-shelf" models. I think there is a similar problem in scientific discovery. There are certain aspects of science that are not domain-specific. For example, certain statistical techniques for testing hypotheses, general hypothesis forms and experimental design principles. However, it is very difficult to read books on the philosophy of science and experimental design, and then proceed to conduct experiments in yeast systems biology. One needs a greater specificity of knowledge.

The same reasoning applies to software approaches in scientific discovery. One of the research questions I would like to begin answering during my PhD studies is just how much of the scientific discovery process we can abstract away from specific domains. To this end I have been exploring

using general-purpose theorem provers rather than bespoke algorithms. So far the performance of the domain-specific algorithms is higher for the specific tasks, which agrees with Dr. Calefato's results. The question of trade-offs between specificity and performance remains, and my view is that there is need for modular design of these systems. Which leads to the next theme, how to manage these software development projects.

2.4 Project management

Because of the nature of scientific discovery, experimental platforms need to be developed in tandem with software. This presents many project management challenges for the software engineers. For a start, projects in scientific labs are rarely defined in such clear terms, and are often open-ended. Exceptions tend to be the very large projects, for example CERN or the James Webb Space Telescope. Commonly, the experimental platforms that labs develop are built up piecemeal, when funding allows or necessity dictates.

Human resources are also a major factor in the development of such systems. It is rare for research groups in natural sciences to have dedicated software engineers within the group. In our group, for example, we are collaborating with a software engineering company in India that has a *pro bono* division that collaborates with academia. Additionally, turnover rate in academic groups can be quite high, it being common for post-doctoral researchers to spend one or two years with a group before moving on, when many of these projects run for much longer.

These factors affect the management of the software systems for automated scientific discovery for a few reasons.

1. The interfaces between software systems and experimental hardware are constantly changing. Firmware versions, proprietary software for lab equipment, "hacky" solutions all complicate this problem.

2. The scope of possible experiments, and therefore the requirements on the system, are also changing.

3. Without specialist support, academic research groups will struggle to have the time or the expertise to develop software systems that would meet the requirements of their domain, particularly with regard to robustness and scalability.

Consequently, we have found it best to employ an iterative approach to project management, where we have focused on achieving small milestones and checking in frequently to see what can be done next. The role of project manager is taken on by various different group members at different times, which can be confusing. This is certainly an area for improvement within our group.

2.5 Devbots for scientific discovery systems

In her guest lecture Linda Erlenhov discussed the use of so-called "devbots" in software engineering. I think these could be extremely useful within scientific discovery because of the relative lack of software engineering skills in natural science research groups. Researchers like me—those who have good domain knowledge and a reasonable skillset when it comes to programming, but no formal software engineering experience—would greatly benefit from bots that would make suggestions on how to structure software systems, or refactor code to make it more efficient and easier to maintain.

2.6 Sustainability (of software products)

Ideally, software systems for automated discovery will be well sustained, allowing for application as scientific knowledge and experimental practices develop, and also in order that results can be replicated and interrogated by other research groups. In reality there are many barriers to this being realised, of which I will explore a few.

One such barrier is the aforementioned limited human resource within scientific groups. Researchers are rarely hired as software engineers, so will have limited time and inclination to properly

maintain tools. The current system in academia would offer limited reward for such behaviour, as reproducibility tests and sharing of tools are not top-tier criteria for many grant applications or research positions.

Another limiting factor is the lack of available standards. This means that even when researchers invest time to achieve a sustainable software platform, they often have to make up their own definitions of what sustainability means. There are growing movements to bring about such standards, for example the COMBINE community within systems biology. Engagement with these communities is far from universal, but they offer the best forum for addressing issues of sustainability within each domain.

3 Future trends and directions of software engineering

I think that machine learning and AI present novel challenges to software engineering, and two major themes need addressing.

The first main theme is data. We talked in the class session about having a requirement for versioning of data, similar to that of versioning software. This is because any machine learning system is characterised and shaped by the data that is used to train it. In order to have sufficient trust in and robustness in the use of machine learning systems, some versioning is necessary. Other related issues are that of data provenance (giving due credit and control to the owners or originators of the data) and data fairness (including ensuring that datasets represent a diverse sample from the underlying distribution).

Secondly, there is the challenge of human-computer interaction. Humans and machines will need common languages, particularly in science when the precision of communication is vital. We cannot place all our faith in natural language processing when it comes to AI researchers. I think that formalisation of these languages will help not only with day-to-day interaction between humans and machines, but also interrogation of AI systems to increase trust and accountability.