CMSC 22100 Research Report, Spring 2019

The research report is a team project designed as an introduction to computer science research in the area of programming languages. In completing this work, you will draw on what you've learned throughout the quarter and see how the formalisms of the theory of programming languages are employed in professional research.

Your two-person team will review in detail a publication from within the realm of programming languages research. Your team will be assigned a paper (or other document) at random from among a pool of significant publications. You should expect to spend at least a week doing a first reading of the paper, talking it over, and trying to understand its meaning. The fact that there are two of you is for you to engage in a dialogue whereby you help one another both reach a satisfactory understanding. Expect to challenge, question, share bewilderment with, explain ideas to and hopefully share epiphanies with your report partner.

To be clear: complete understanding of every detail of your paper is not the goal of this report. The goal is for you to reach a good working understanding of the research such that you can present its main contributions clearly to a knowledgeable audience and you can answer basic questions about it, without needing to be able to illuminate every dark corner.

There are be two components to the research report: a written document ("the document" hereafter), and a presentation before your classmates. We will have several presentation sessions towards the end of the quarter, dates and times to be announced; there are too many of you for us to do all the presentations on the last day of class alone. One of these sessions will be during finals week.

Here is a suggested outline for the document. Each enumerated item is discussed in greater detail below.

1. a high-level summary of the work for a general audience,
2. more technical summary suitable for a knowledgeable audience,
3. a digest of any related work you explored in preparing the report,
4. a report on any *constructive engagement* with the work, and
5. a survey of the current status of the work.

The high-level summary of the work (item 1) should state as clearly as possible what problem the work addresses and name how it does so, without explaining the details. For example: "The programming language designed in this paper is designed for building working schedules so that no worker is ever needed in two places at once, and every worker always has enough time between tasks to get from one work location to another. The authors have designed a system of 'schedule types' in order to achieve this goal." This section might include a few examples to assist in understanding. You should have in mind that you could show this part of the document to any computer science major on campus, and they would have some idea what it meant.

The technical summary (item 2) is where you should include details about how the system actually works. Definitions of term grammars, typing judgments, and evaluation relations would be appropriate in this section, as would discussions of scanners, parsers, typecheckers, compilers, theorems, lemmas, mechanized metatheory, *etc.* You should not recapitulate the entire work of the original publication, but rather select a few characteristic and interesting bits. You should have in mind that you are writing this section for one of your 22100 classmates, who shares with you a certain technical vocabulary and set of related experiences.

In reviewing your publication, you may find that there are certain obstacles to your understanding. For example, the authors might refer to "Haskell-style type classes" at several key points; you might not know what that means. In those cases, you should peruse the bibliography and see if there are some materials you could refer to for understanding this missing piece.[1] Whatever additional materials like this you consult in preparing your report, you should write about in item 3.

Item 4 is about getting your hands dirty. It may be the case that your publication refers to systems you can download and use yourself. For example, if your paper is all about the Rust language, you should install the Rust compiler and write some Rust programs. If your paper is about Haskell, and you don't know Haskell, then learn some Haskell. Similarly, if your paper outlines a formal system (a grammar/typing/evaluation trifecta, as in TaPL) you might try to write part of a compiler for it (in which case expect that you might uncover typos and formal gaffes). This sort of constructive investigation can be enormously helpful as a reader. Think of how well you understand, for example, how "if", "and" and "or" work in programs now that you've written small-step and big-step evaluators in recent weeks. Write about whatever you've done along the lines of constructive engagement in item 4.

Finally, for item 5, you should spend some time learning about what the current status of the work is at the time of this writing. If it describes a programming language, is that language broadly popular among programmers or academics? If it hasn't achieved widespread popularity, is it used in a certain kind of programming or a certain kind of research? Is there a user community around it, and documentation? Is it possible to download and install an implementation of a related system for your machine? Has it become part of some other system; for example, has the main idea been absorbed into Java or Ruby? Has the

[1] Please note that some papers are *much* clearer than others, regardless of technical merit; not all papers are written equally well!

system faded into obscurity? Can you find no mentions of it subsequent to its publication? This is what belongs in item 5.

The document must be prepared with LaTeX and must include a bibtex bibliography. Expect to write about five pages. We will provide a template for you as a starting point.

The presentation will consist of a five-minute summary of the work, conveying its main features and calling out a few interesting details of it, followed by three minutes for questions. In order to accommodate all of you, we will need to adhere to this time frame strictly. You may prepare slides if you choose; you do not have to. You may also present interactive demos with advance notice. Note that it is possible to do perfectly well with no more than chalk and a blackboard. A warning that if you have more than four or five slides, you are unlikely to have enough time to present all of them in five minutes. If you do prepare slides, they must be in pdf format and you must send them to Professor Shaw attached to an email to ams@cs.uchicago.edu no later than twenty-four hours in advance of your presentation. All slide presentations will be copied onto and run from the same (Ubuntu) laptop at presentation time.

Your team will receive one grade; there will be no distinction between what one person earns versus the other. It is up to the two of you to divide the work in some mutually agreeable way.

You will both submit in a directory entitled "research_report" in your git repository. One student's directory should contain the actual report and a slide presentation if you prepare one, and the other's should contain exactly one file named "partner.txt" containing the name of your report partner. This way, there will only be one actual submission per team.

You may petition to report on a different paper rather than your assigned one. No petition may be made until at least one week has passed after the random paper assignment. Instructions on how to do so will be posted on piazza a bit later.

Please note: graduating students should pair with other graduating students if possible. If any non-graduators pair with graduators, both students will need to adhere to the graduators' earlier deadlines. If there is some reason we cannot form pairs for everyone, because the class has an odd number of students, or we encounter some unforeseen difficulties (such as withdrawals), we will make adjustments as needed. It might be the case that some students have to work alone, especially if there are withdrawals, although this is not intended and we will do our best to arrange for it not to happen.