Seminar on Computer-assisted mathematics

Judith Ludwig, Florent Schaffhauser and Junyan Xu

Heidelberg University - Summer semester 2025

Session 5 - May 22, 2025

Searching Mathlib

Today we'll learn about searching Mathlib.

Broadly, there are two kinds of search tools:

- Exact search: Mathlib4 documentation, Loogle, and code search
- Fuzzy, semantic search: LeanSearch and LeanExplore

Mathlib4 documentation (docs)

- The game is to guess the name of the declaration (`theorem`/`lemma` or `def`) that you want to find. To guess correctly more often you may want to familiarize yourself with the Mathlib naming conventions. The name usually includes the most important declarations ("constants") mentioned in the statement.
- The search functionality of the mathlib4 docs matches possibly non-consecutive occurrences of the characters in the input string in the particular order. Lowercase letters can match the corresponding uppercase letters, but not vice versa.
- For example, searching for `aeval` will find `minpoly.aeval`, `Module.AEval`, but also `RatFunc.eval`. Searching for `Aeval` will find `Module.AEval` but not the other two.
- The ordering of the search results is not ideal; `Polynomial.aeval` doesn't appear in the 30 search results even though it's more basic than (in fact used in) `minpoly.aeval`.

Mathlib4 documentaion (docs)

- Apparently it doesn't match docstrings, or maybe there is too much noise so it is not an effective way to search docstrings.
- Updated three times a day to the latest Mathlib4 master branch.

Loogle

- Loogle is the choice if you are looking for declarations relevant to one or a set of declarations, e.g. when you are working on a particular goal and want to find `lemma`s that can reduce it to simpler goals or `def`s that constructs data of a particular type.
- For example, searching for `AlgHom, Polynomial` will return `Polynomial.aeval` as the second result.
- If you think a string of characters is likely to occur in the declaration name, you might include them as a keyword (case insensitive) using quotation marks: searching for `minpoly, "unique"` will return `minpoly.unique`, for example.
- Can be used in VSCode (but still requires Internet connection): just type e.g. `#loogle AlgHom, Polynomial`.
- For offline use, see Zulip discussion.

Code search

VSCode / GitHub search

- Matches consecutive characters by default, including in docstrings; regular expressions are also supported.
- No proper ordering of search results because VSCode/GitHub aren't aware of the dependency between declarations.
- For example, if you are not sure about the proper syntax to invoke the tactic `wlog`, searching for `wlog` will give you code samples.
- You can also search pull requests and issues with GitHub search.

Semantic search

- LLM-powered, retrieval-based
- Current services include LeanSearch, LeanExplore (new from May 9th), and Moogle (not up-to-date)
- Use when you are interested in some mathematical result but are not sure in what form it could be stated in Mathlib, not even what constants the statement might involve.
- Example: searching `law of cosine` in LeanSearch.

Tracing Mathlib history: Mathlib changelog

- Mathlib is a large collaborative project with many contributors. It saves time and effort if you can connect to people with expertise on a part of the library / codebase that you are working on.
- Mathlib changelog makes it easy to find out when (in which pull request / commit) a declaration is added, changed, or removed.
- Useful if you want to go to the pull request to find out the responsible author(s) to contact and discussions (or links to Zulip discussions) about design choices, etc.

Tracing Mathlib history: Git blame

- Git itself tracks history pretty well, and allows you to find out who made changes to a particular line of code. This info can be shown right next to the code using GitLens, in command line via `git blame`, or via GitHub's interface:
- Go to an arbitrary file on GitHub, and then press `B`, you'll see when and by whom each block of code was last changed. There is a button you can press to see the state before the change, so you can trace the history recursively.
- It may lose track if contents are moved across files (e.g. when a file is split into multiple parts), and in such case you need to go to the file containing the declaration before the move to continue tracing.
- Some old declarations trace all the way back to the archived Mathlib3 repo (also check out Mathlib3 changelog).

Searching for tactics

Tactic cheatsheet: useful for figuring out whether there is a tactic for a specific task. Includes many tactics that we don't have time to introduce. Tactics missing from the cheatsheet include `rintro` (combining `intro` and `obtain`, can also perform substitution when fed `rfl`). (`rcases` is subsumed by `obtain` and the code is shorter with `obtain`.

Commands missing from the cheatsheet include `#find_home` and `#trans_imports`, though only relevant when you want to contribute to Mathlib. If `#find_home <declaration_name>` tells you a proper file to put your result, great. Otherwise, you can experiment with adding imports to an existing file and `#trans_imports "Mathlib."` tells you how many transitive imports are introduced.

The tactics pages of the Mathlib Manual are more detailed.

Projects

Projects are now available in the main branch.

Judith's projects: Principal Ideal Domains, Noetherian Rings

Florent's projects: Affine Isometries, Polynomial Function

Junyan's projects (roughly from easiest to most difficult):
PrimitiveRoots, CoinProblem, Kuratowski, CoinTheorem, degree_ratFunc
All of these shall be made pull requests to Mathlib when completed.

Assignment

Our next session is on June 5th. Your assignment for next week is:

1. Choose a project.
2. Get started. Make sure you understand the maths first.