# OCamlOScope – New OCaml API Search

Jun Furuse

May 24, 2014

## 1 OCamlOScope – New OCaml API Search

OCamlOScope (`http://ocamloscope.herokuapp.com`) is a new search engine for OCaml programming. Tons of OCaml library packages, modules, types, constructors, functions and values can be searched via simple string queries.

The talk will be to advertise this new search and show its technical details, demonstrating OCamlOScope. It would be great if the audience tries it and give me some feedback and suggestions.

## 2 Search engines for typed functional languages

Actually, searching library functions and other things by their name or types were well known technology: there are old academic researches[1]. In Haskell, Hoogle(`http://www.haskell.org/hoogle`) is a programming framework pretty commonly used .

OCaml has such tools already, too. OCamlBrowser has existed in OCaml compiler source code for long time since Objective Caml 3.00. OCaml API Search(`http://search.ocaml.jp/`) was a port of OCamlBrowser as a CGI based web service.

## 3 Why OCamlOScope?

Unfortunately, OCamlBrowser and OCaml API Search are not really useful these days. They have drawbacks:

- OCamlBrowser is a local tool: it does not show you things you do not install.
- OCaml API Search only targeted at OCaml standard library and Extlib, which are now considered pretty limited, with other rising of 3rd party libraries.
- It would be possible to add more libraries to OCamlBrowser/OCaml API Search, but pretty likely it is not very scalable. The data base is a collection of cmi files and they are loaded at each query from the disk.
- Impossible to search against multiple libraries which contain modules of the same names.
- Their search algorithm is too simple to find what you want from imperfect name and type information in your mind.
- Their algorithm cannot work well against things defined in local modules.

OCamlOScope, a successor of OCaml API Search, tries to overcome these issues.

- It is a web service but also possible to install locally.
- It aims to search lot more OCaml libraries including Batteries, Core, etc.

- All the data are in memory (so far) for speed. They are also hash-consed.
- The search algorithm is entirely rewritten: it can find fuzzy matches using edit distance like Hoogle.

## 4 Technical details of OCamlOScope

DB creation:

- OPAM and Findlib(OCamlFind): Search DB is constructed per OPAM package basis. Search results are displayed with their Findlib package names.
- -binannot: The DB creation is hugely dependent on cmt/cmti files.
- Source code documents are obtained by running OCamlDoc using compilation commands recorded in cmt/cmti.

Scalable search:

- OCaml compiler's paths and types are converted down to more compact data and hash-consed for in-memory database.
- Search with edit distance based thresholds to allow partial matches and reduce recursive visits of type trees.
- Search cache to avoid repeating searches with the same query, useful when users visit multiple result pages of the same search.

Visualization:

- Ocsigen/Eliom
- (The address is herokuapps.com but it is no longer a heroku service.)

## 5 OCamlOScope is NOT a simple Hoogle clone : Incoming features, or Issues to be solved

OCamlOScope is very useful already, but I must say that it is still far from perfection. Running it against large libraries like Batteries and Core, we found some OCaml specific issues: OCamlOScope cannot be a Hoogle clone.

We need good grouping of search results. OCaml has SML style module system with module aliases and inclusion. They are used pretty common in many modern OCaml libraries. As a result, you see tons of look-alikes in OCamlOScope search results, which are aliased multiple times by them. This may also impact on scalability: we should be able to reduce number of search items significantly by grouping aliased items.

Currently the search results are grouped depending on their "looks" (textual representations) but it helps, but is not a correct approach. I am going to work on solving this by tracking module aliases, something I have similarly done in OCamlSpotter(`https://bitbucket.org/camlspotter/ocamlspot`). It is not yet done but I hope I could show some results at the talk if the proposal is accepted.

There are some other todos listed, some maybe solved till OCaml 2014:

- Portable DB files: currently DBs are not portable and cmt/cmti's must be compiled in the computer which runs OCamlOScope
- Jump to source
- Search in documents
- T-shirts :-)

## References

[1] Mikael Rittri. *Using types as search keys in function libraries.* Journal of Functional Programming, 1(1):71–89, 1991.