SearchSECO: A Worldwide Index of the Open Source Software Ecosystem

Slinger Jansen
Utrecht University, CWI

Siamak Farshidi
UvA

Georgios Gousios
TUDelft, Facebook

Tijs van der Storm
Groningen, CWI

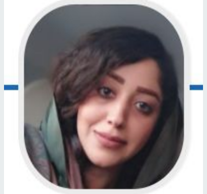
Joost Visser
Leiden Uni

Magiel Bruntink
SIG

Tom Peirs
Research Assistant

Swayam Shah
Research Assistant

Floris Jansen
Research Assistant



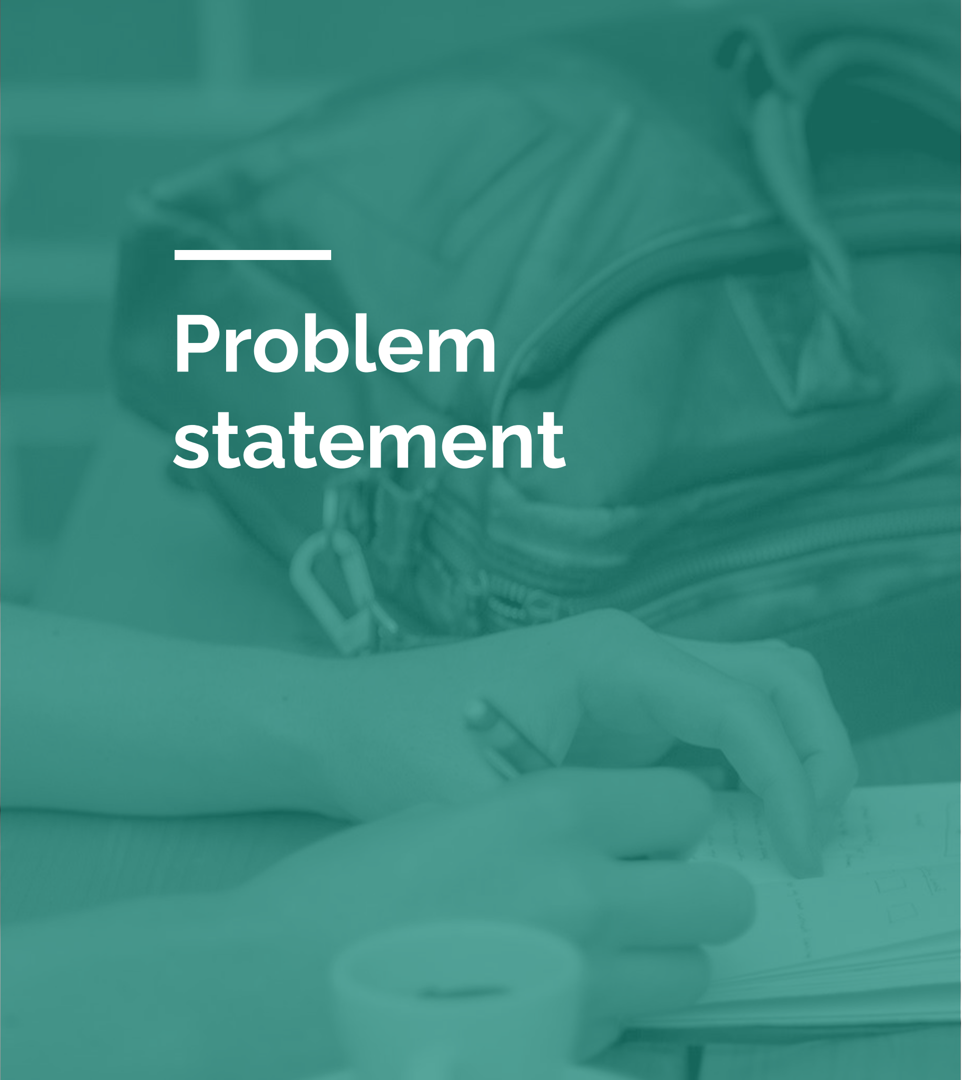
Elena Baninemeh
Research Assistant

Jozef Siu
Research Assistant

Fang Hou
Research Assistant

Donny Groeneveld
Research Assistant

Siamak Farshidi
Senior Researcher



Problem statement

It is possible to find source code in the software ecosystem at

- file level (SHG) or
- line level,

but not at *method level*.

Our Suggested Solution: SearchSECO

SearchSECO maintains an index of all source code in the worldwide software ecosystem.

1. We continuously **spider the software ecosystem** for source code.

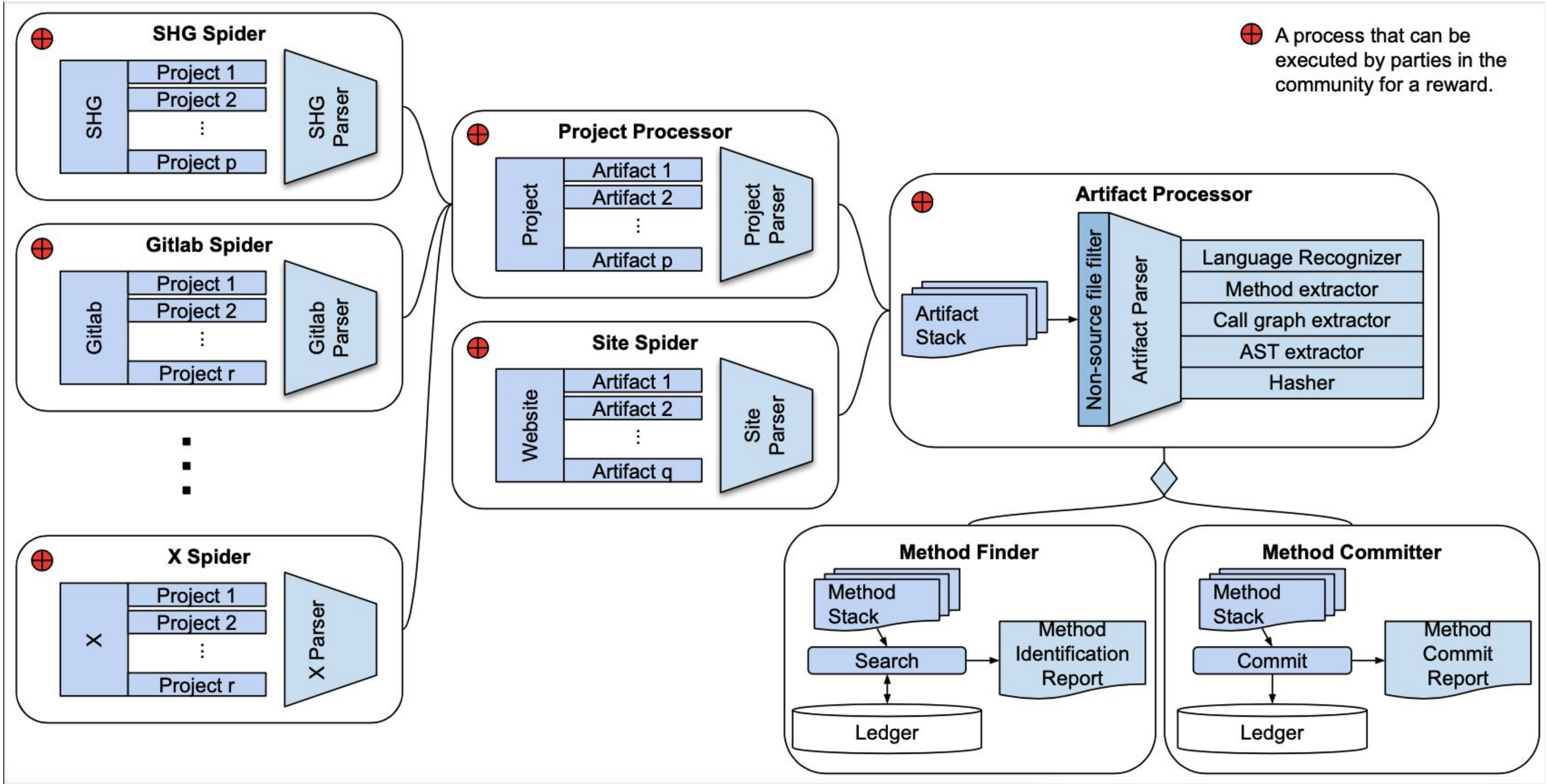
2. We **extract the abstract syntax tree** and hash it for quick access and search.

3. We **annotate the source code** and store it for posterity.

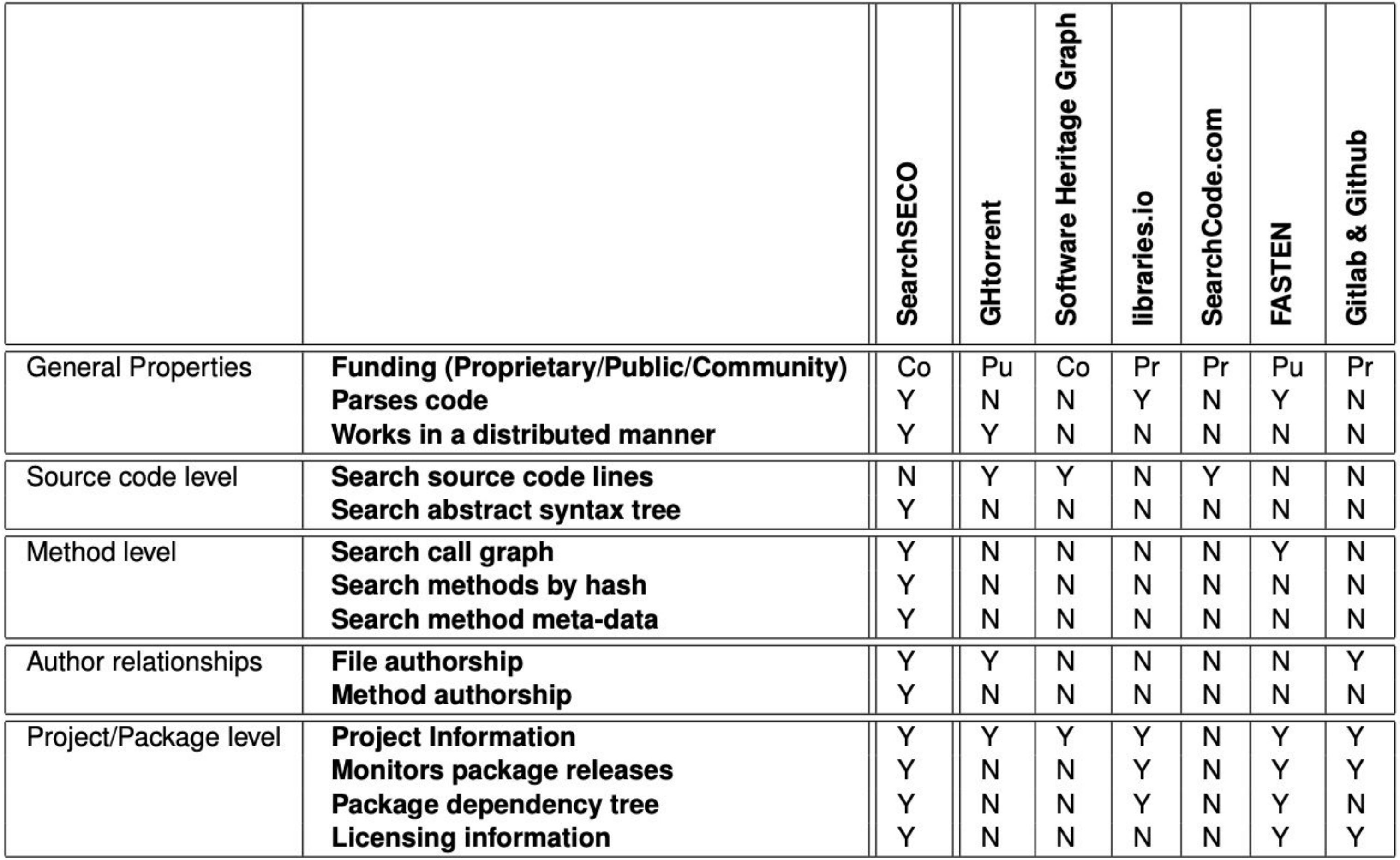
4. We provide a search engine for **worldwide method and AST search**.

5. We **create and analyze models** for making the software ecosystem a safer place.

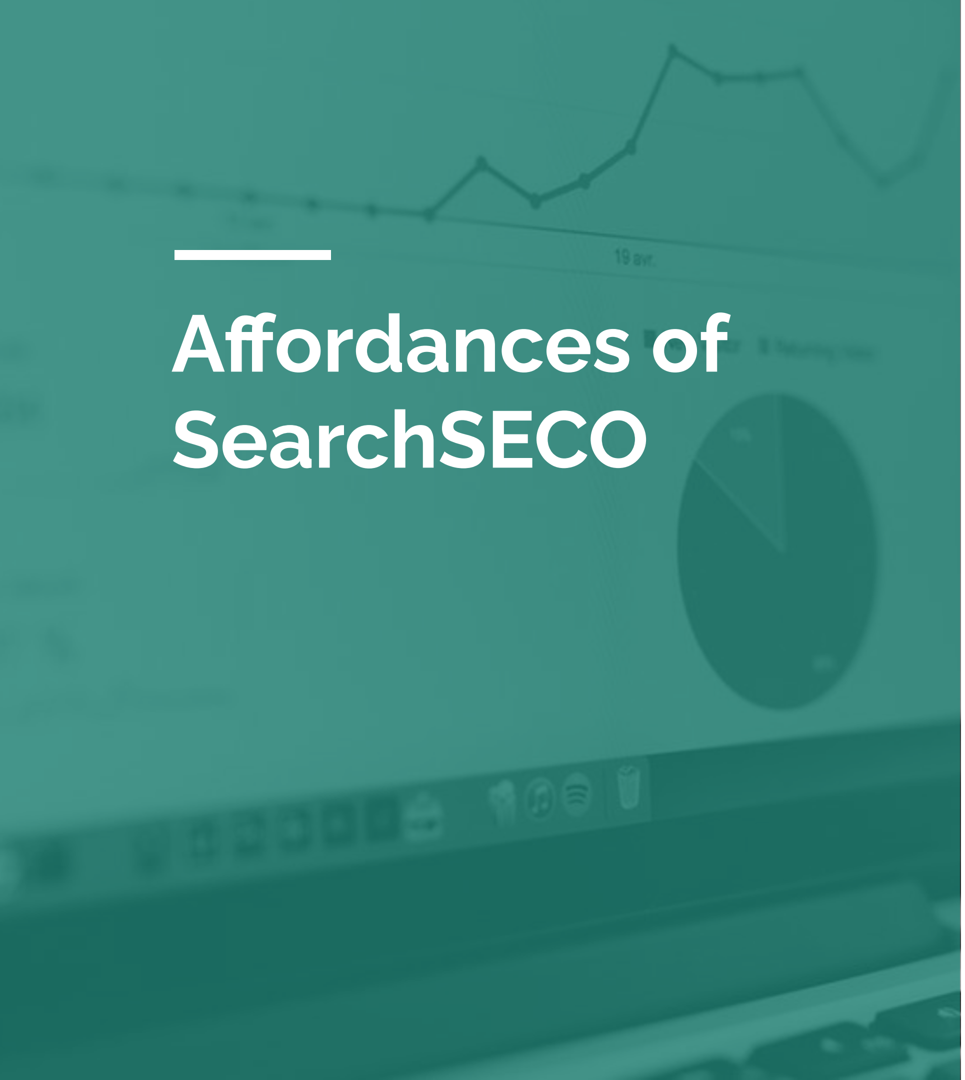
We spider the worldwide software ecosystem



How do we compare to other source code search engines?



		SearchSECO	GHtorrent	Software Heritage Graph	libraries.io	SearchCode.com	FASTEN	Gitlab & Github
General Properties	**Funding (Proprietary/Public/Community)**	Co	Pu	Co	Pr	Pr	Pu	Pr
	Parses code	Y	N	N	Y	N	Y	N
	Works in a distributed manner	Y	Y	N	N	N	N	N
Source code level	**Search source code lines**	N	Y	Y	N	Y	N	N
	Search abstract syntax tree	Y	N	N	N	N	N	N
Method level	**Search call graph**	Y	N	N	N	N	Y	N
	Search methods by hash	Y	N	N	N	N	N	N
	Search method meta-data	Y	N	N	N	N	N	N
Author relationships	**File authorship**	Y	Y	N	N	N	N	Y
	Method authorship	Y	N	N	N	N	N	N
Project/Package level	**Project Information**	Y	Y	Y	Y	N	Y	Y
	Monitors package releases	Y	N	N	Y	N	Y	Y
	Package dependency tree	Y	N	N	Y	N	Y	N
	Licensing information	Y	N	N	N	N	Y	Y



Affordances of SearchSECO

Relationships between **methods**

- Study method co-evolution across projects
- Weaknesses tracked, fixes propagated

Relationships between **authors**

- Fine grained authorship
- Copy-paste behavior (StackOverflow)

Relationships between **software projects**

- Establish package dependencies and cohesion
- License violations

RC1: Mining the Worldwide Software Ecosystem

We develop a job scheduler that maintains its list of jobs to be done

Worker nodes can pick up these jobs in the ecosystem, similar to the CrossFlow [1]

Examples of automated tasks are:

- Spider an existing project repository for updates
- Extract code fragments from Stackoverflow
- Parse a new project and identify the languages used
- Send out alerts to owners of encountered code fragments
- Check whether evidence of a code fragment still exists

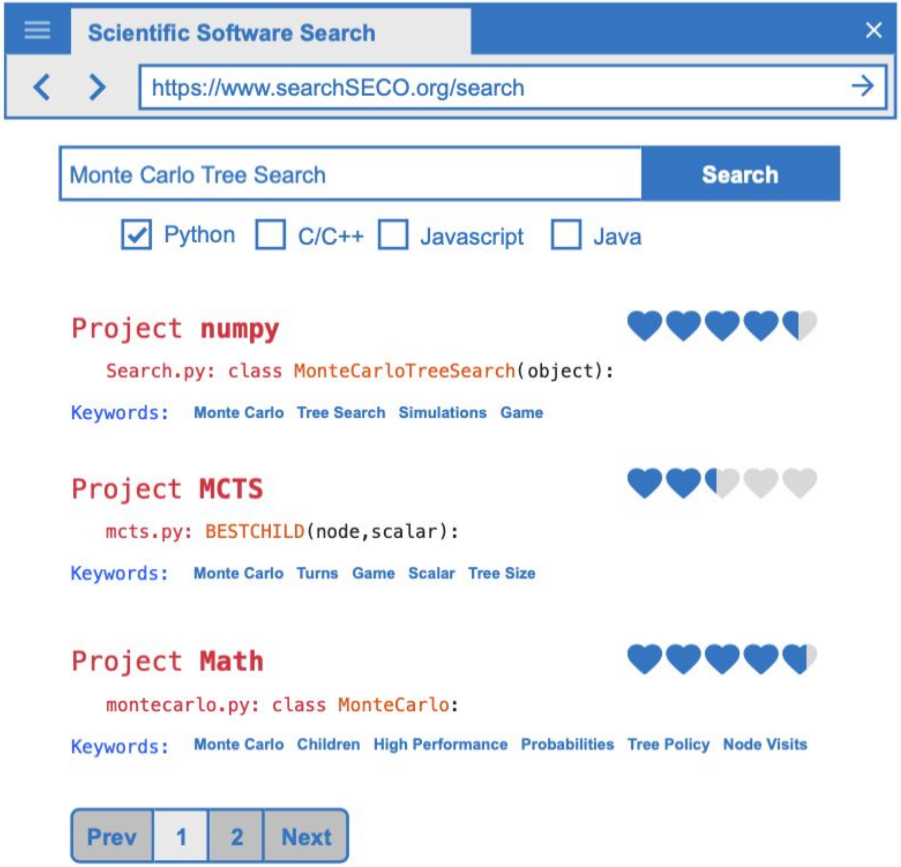
Automated tasks will be incentivized to ensure positive contributions to the community.

Some jobs may stay in the scheduler for a long time; as for these jobs, the correct parser may not yet be available.

In this way, we can easily prioritize which parser needs to be most urgently built.

[2] Kolovos, P. Neubauer, K. Barmpis, N. Matragkas, and R. Paige, “Crossflow: a framework for distributed mining of software repositories,” in 2019 IEEE/ACM 16th International Conference on Mining Software Repositories (MSR) IEEE, 2019, pp. 155–159.

We make worldwide software ecosystem searchable



Scientific Software Search

https://www.searchSECO.org/search

Monte Carlo Tree Search | Search

☑ Python □ C/C++ □ Javascript □ Java

Project numpy

```
Search.py: class MonteCarloTreeSearch(object):
```

Keywords: Monte Carlo Tree Search Simulations Game

Project MCTS

```
mcts.py: BESTCHILD(node,scalar):
```

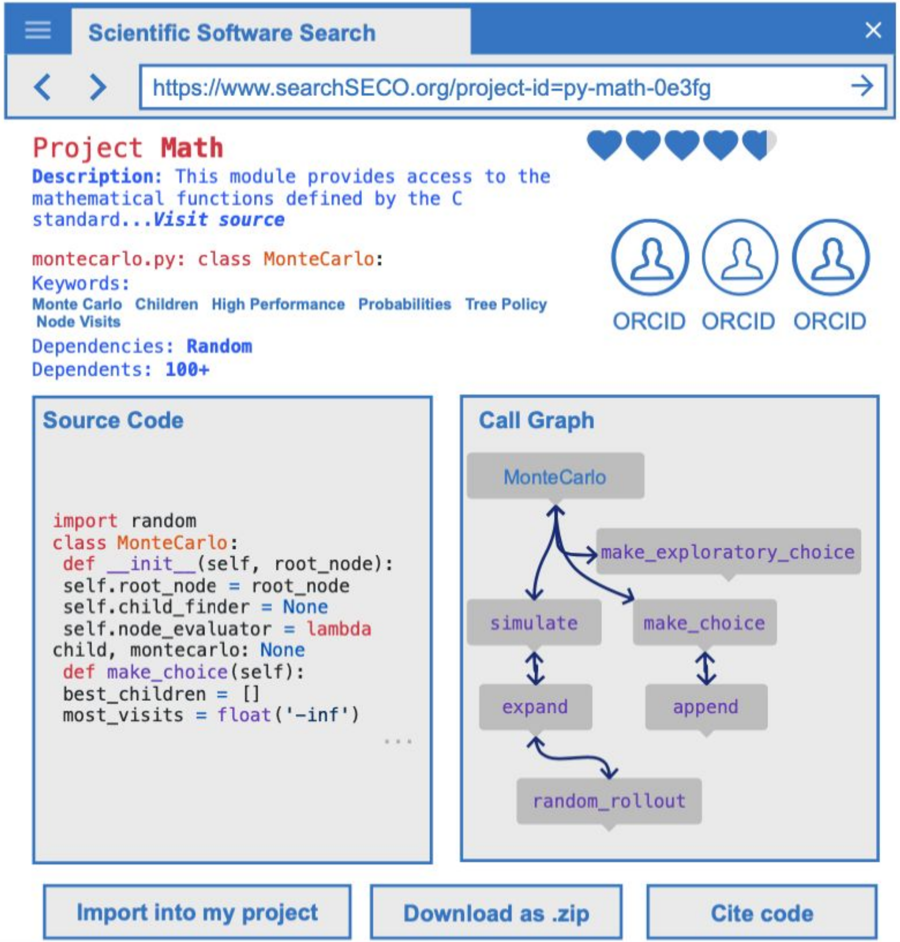
Keywords: Monte Carlo Turns Game Scalar Tree Size

Project Math

```
montecarlo.py: class MonteCarlo:
```

Keywords: Monte Carlo Children High Performance Probabilities Tree Policy Node Visits

Prev | 1 | 2 | Next



Scientific Software Search

https://www.searchSECO.org/project-id=py-math-0e3fg

Project Math

Description: This module provides access to the mathematical functions defined by the C standard...*Visit source*

```
montecarlo.py: class MonteCarlo:
```

Keywords:
Monte Carlo Children High Performance Probabilities Tree Policy Node Visits

Dependencies: Random
Dependents: 100+

ORCID ORCID ORCID

Source Code

```
import random
class MonteCarlo:
 def __init__(self, root_node):
 self.root_node = root_node
 self.child_finder = None
 self.node_evaluator = lambda
child, montecarlo: None
 def make_choice(self):
 best_children = []
 most_visits = float('-inf')
                                   ...
```

Call Graph

Import into my project | Download as .zip | Cite code

RC2.1: Parsing Worldwide Software Ecosystem

Language parametric clone detection

Currently, we have parsers available for Java, C, and C++ from

- FASTEN https://www.fasten-project.eu/
- srcML https://srcml.org
- Rascal https://www.rascal-mpl.org/

Hashing the Worldwide Software Ecosystem

We use VUDDY [1], a high performance method search that hashes ASTs and method signatures for C.

We extend the technology to include other languages (js?)

[1] Kim, S., & Lee, H. (2018). Software systems at risk: An empirical study of cloned vulnerabilities in practice. *computers & security*, *77*, 720-736.

RC2.2: Generic Models for Cross-lang Dependencies

We use these models to track dependencies induced by call-graphs and other relations (e.g., inheritance).

Rascal already supports the extensible M3 model [1], for single-language source projects.

We will extend this to support modeling source code facts across different programming language SECOs, as well as the representation of metadata such as authorship, provenance, and versioning.

[1] Basten, M. Hills, P. Klint, D. Landman, A. Shahi, M. Steindorfer, and J. Vinju, "M3: a General Model for Code Analytics in Rascal," in Proceedings of the first International Workshop on Software Analytics, SWAN, 2015.

RC2.3: AI-assisted development of robust, extraction-oriented parsers

Developing parsers for full programming languages requires significant effort.

We will investigate new AI-based techniques to **(semi-)automatically derive parsers** using a combination of grammar inference techniques and corpus analysis.

These parsers might not be accurate enough for developing a compiler but will be sufficiently fine-grained to **extract function bodies and identify call sites**.

RC2.4: ``Diff''-based parsing and extraction

Parsing and analyzing the code of software projects from scratch will not scale.

Instead of parsing/analyzing full source files, these techniques will analyze the difference between versions of files (e.g., as derived from Github) and incrementally update the SMKB.

Explore how methods mutate for AI assisted mutation prediction.

RC3: AI Assisted Graph Mining for Vulnerabilities

- Structure known vulnerabilities from VulnCode (https://www.vulncode-db.com/) so that the vulnerabilities and permutations of such vulnerabilities can best be found in our code database.
- Establish ways to automatically propose fixes and alert code maintainers.
- Pattern-based graph searches that can be used to detect malware.

As the fixes in vulnerability databases are typically well structured and relatively easy to fix, we could automatically generate pull requests for the code to be fixed.

Furthermore, if the tooling developed in this project is adopted widely, we could warn about vulnerable code at the time of a commit.

Discussion

SearchSECO does not make the SMKB a surveillance instrument, we must use design principles that do not easily link software engineers to their identity information.

Can we use data access as a method to incentivize scientists to contribute?

How should we store the database? Distributed? Its estimated at 250TB.

Concluding

- **SearchSECO**: A Worldwide Index of the Open Source Software Ecosystem
- We extract, parse, and store all the source code in the world
- We create a shared infrastructure for researchers worldwide

What do we need? **Feedback and Funding**! When do we need it? **Now**!
slinger.jansen@uu.nl