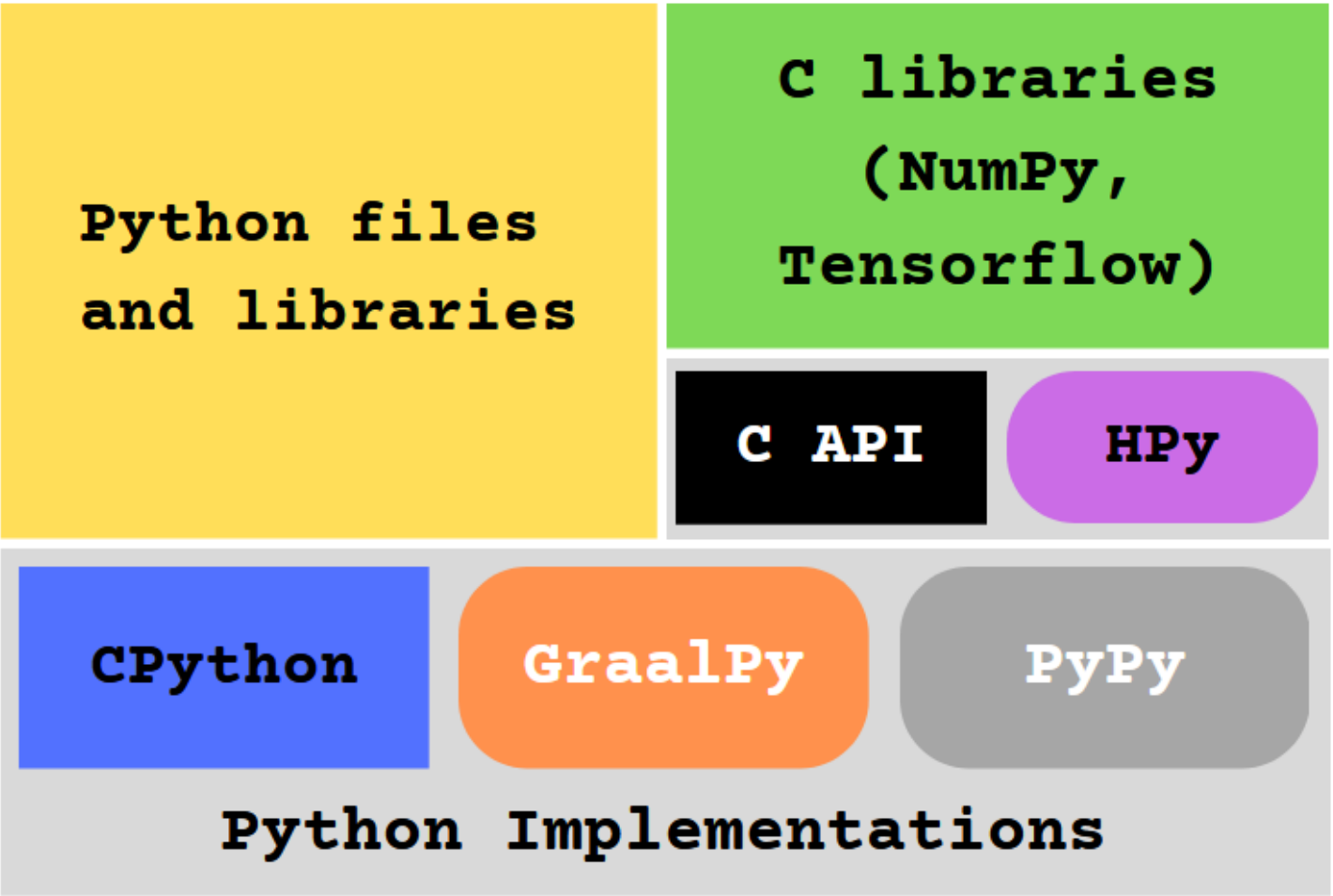
Implementing an HPy Backend for Cython: A Performance Benchmark Study

Du Toit Spies

PyConZA 2024

4 October 2024

Map of Python

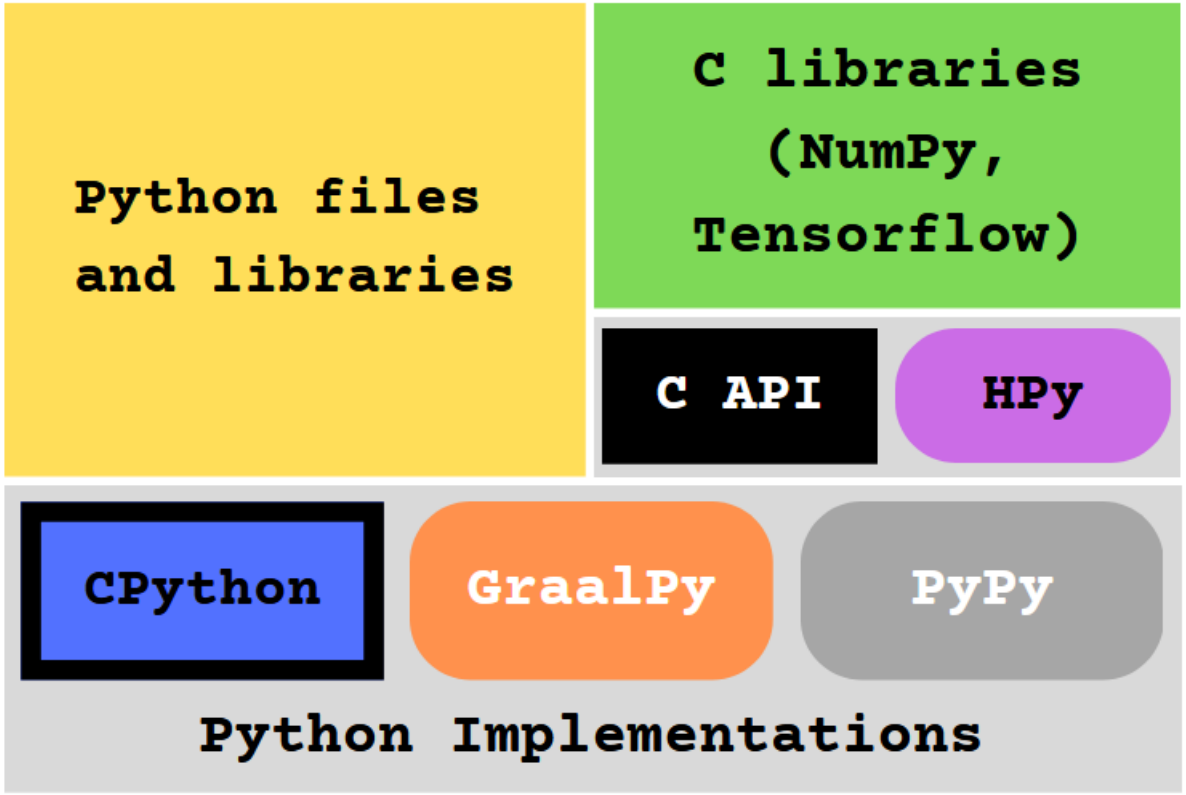


Python Implementations

- Programs that can run Python code correctly
- Features:
  - Different memory management strategies
  - Only Interpreted *vs* Only Compiled *vs* Just-In-Time Compilers
  - Different memory layouts
- This research focuses on **CPython** and **GraalPy**

CPython

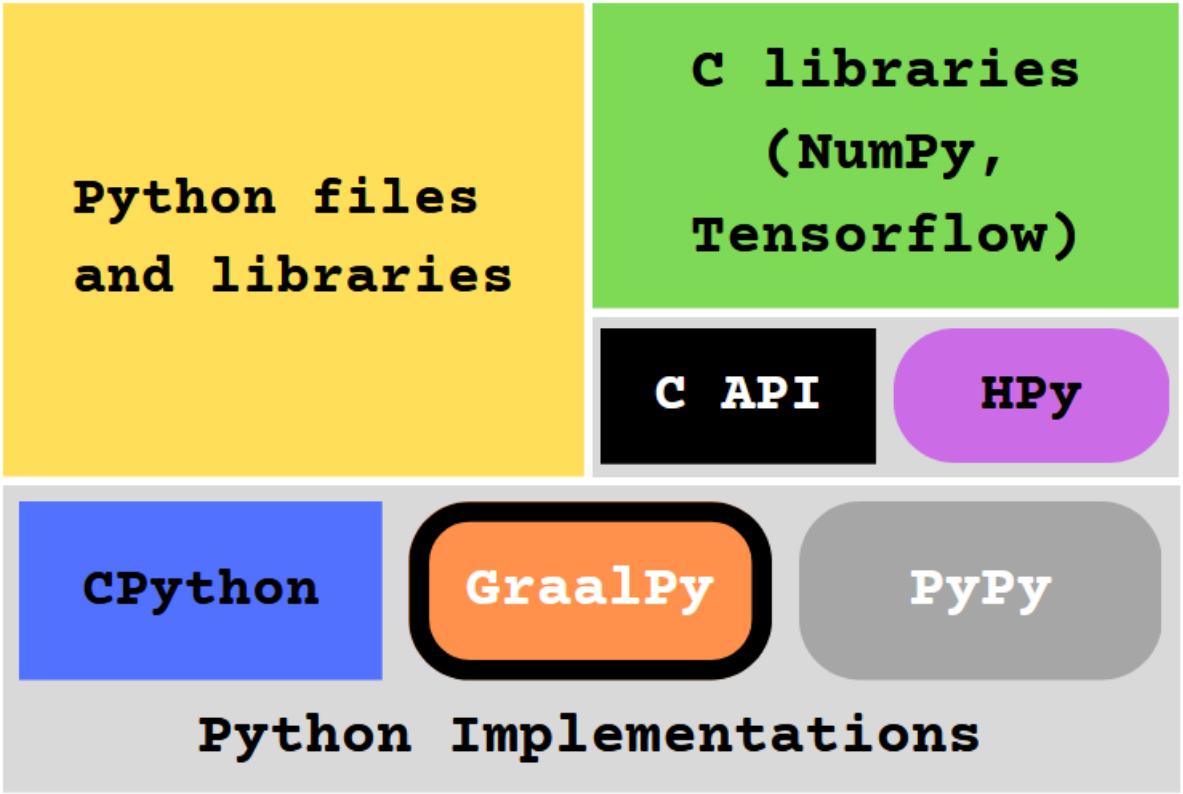
- Reference Implementation
- Written in C
- Only interpreted
- Reference Counting
- Exposes the C API



https://www.python.org/

GraalPy

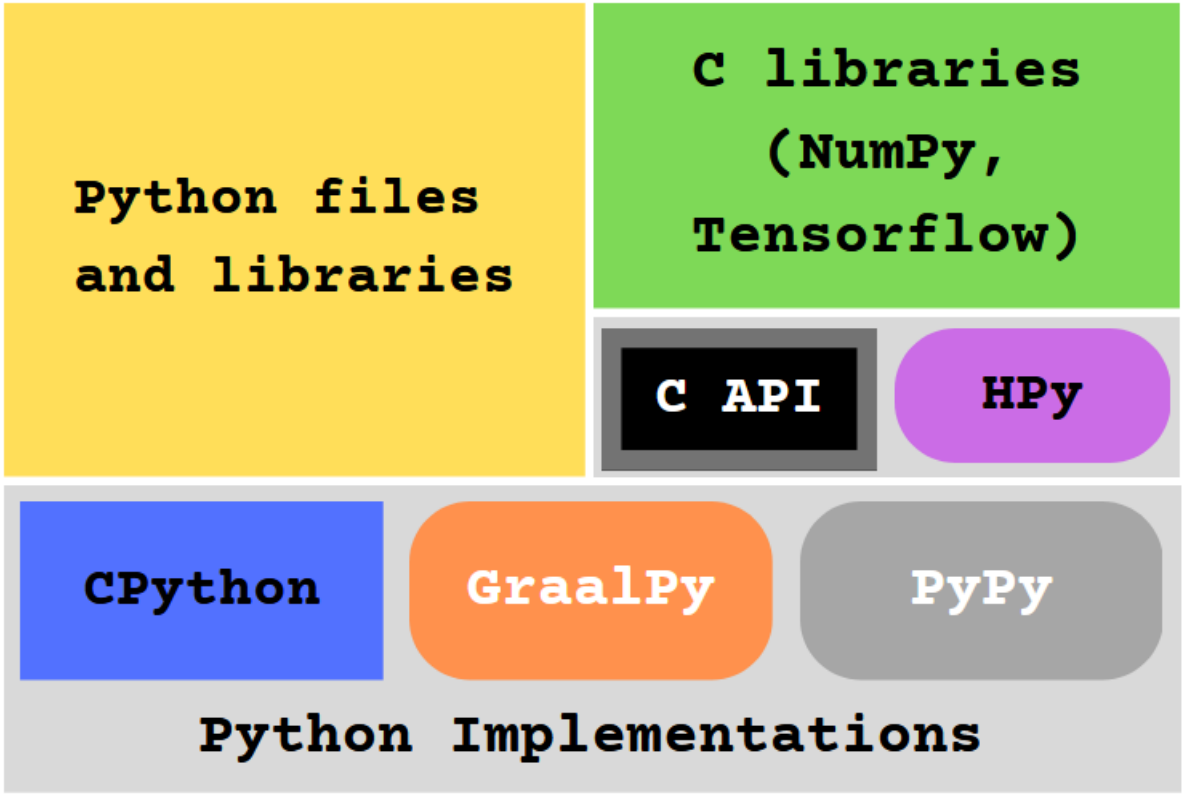
- Java-based implementation developed by ORACLE
- Part of larger GraalVM project
- JIT compiler
- JVM memory management



https://www.graalvm.org/python/

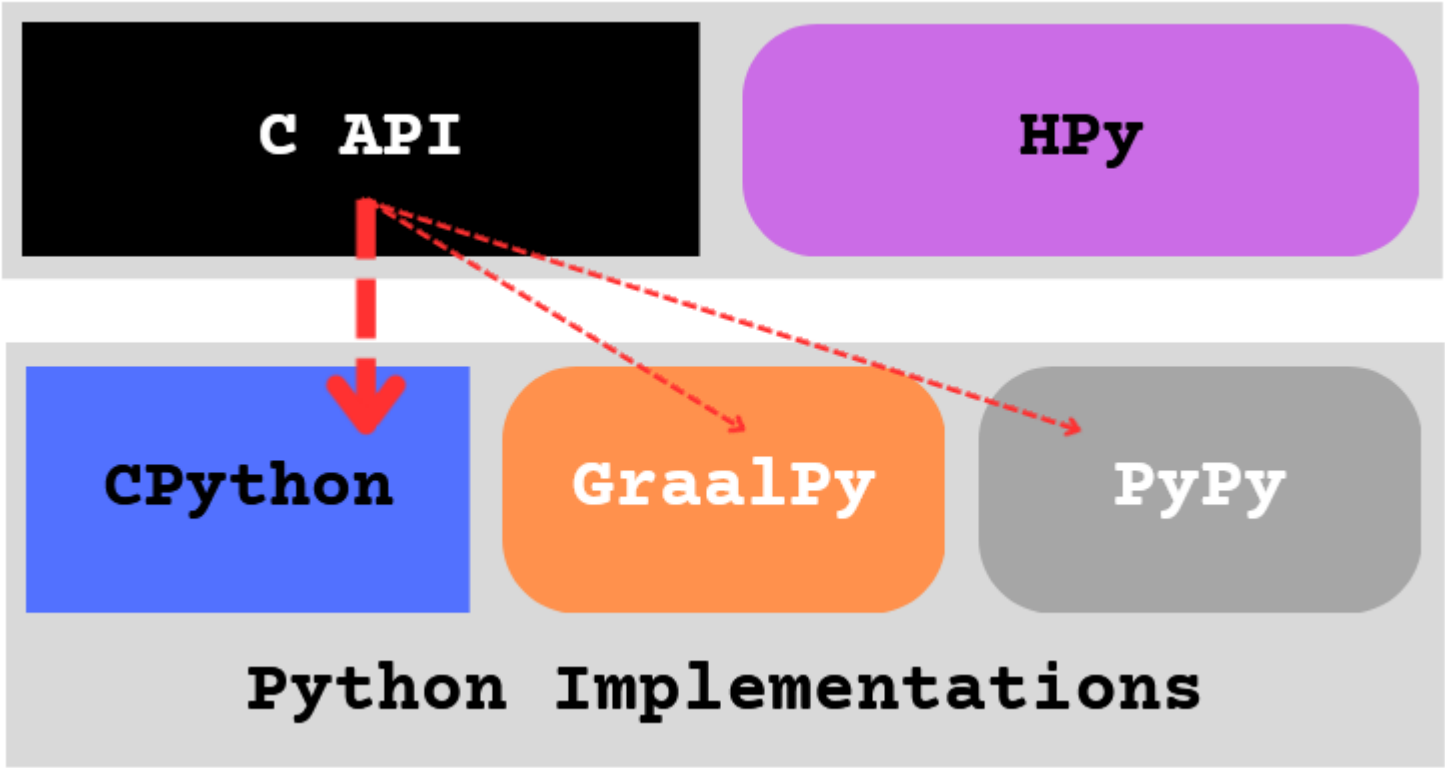
C API

- API that connects C code to the Python interpreter
- Used by many popular extensions
- Exposed by CPython
- Contains many implementation details of CPython



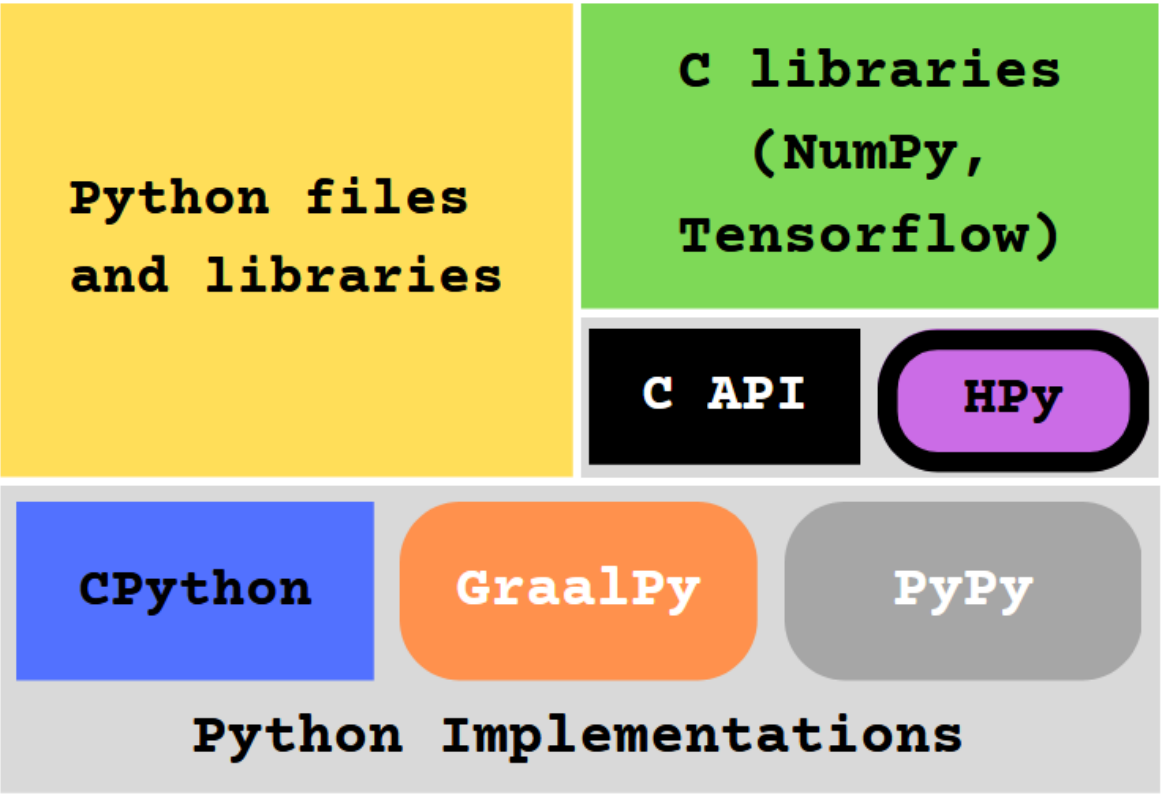
https://docs.python.org/3/c-api/index.html

C API



HPy

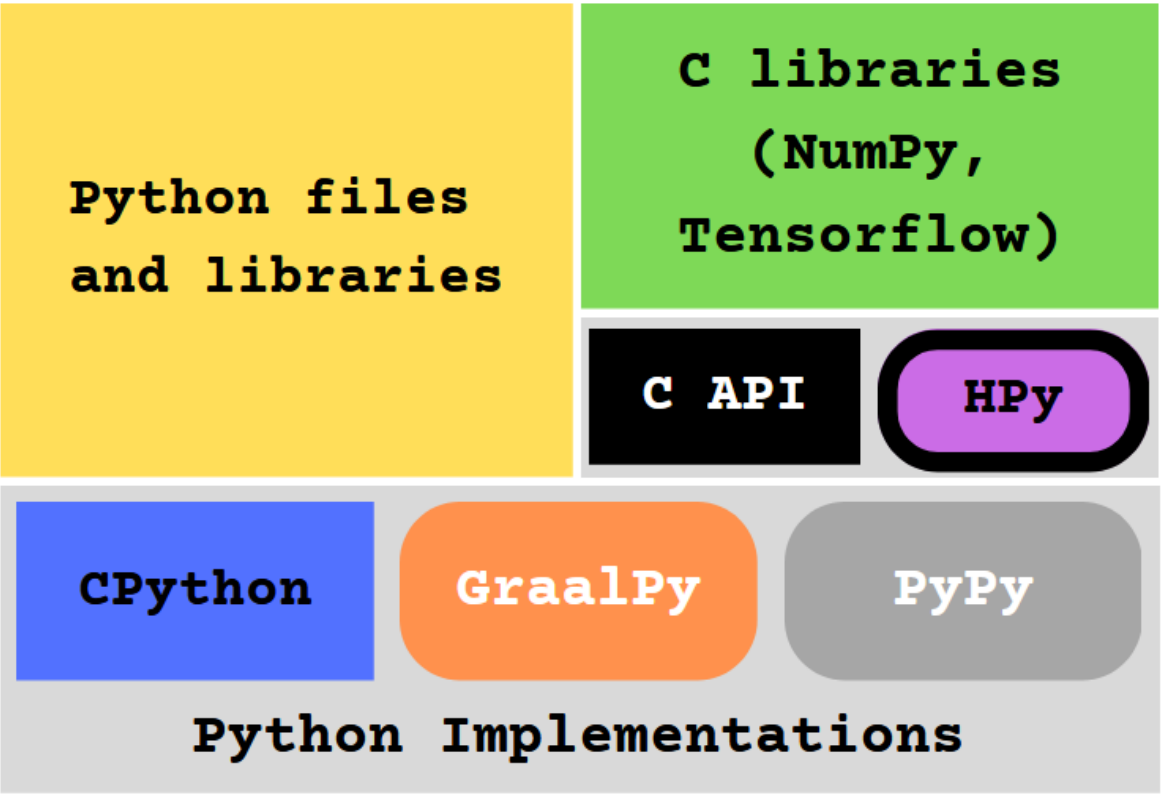
- Alternative to the C API
- More generic – not as tied to CPython as the C API
- Runs much faster on implementations like GraalPy
- Aims for no performance loss on CPython
- Not extensively benchmarked yet



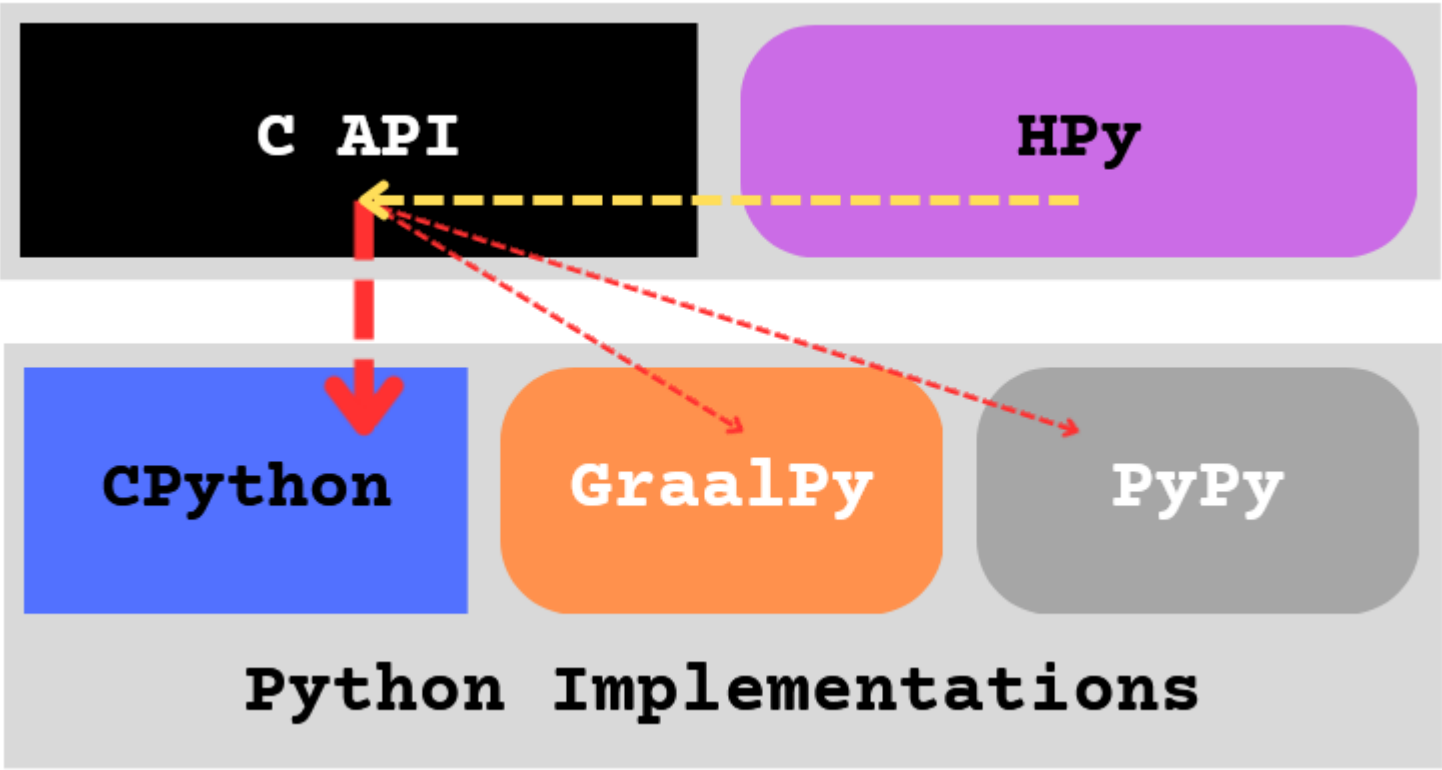
hpyproject.com

HPy

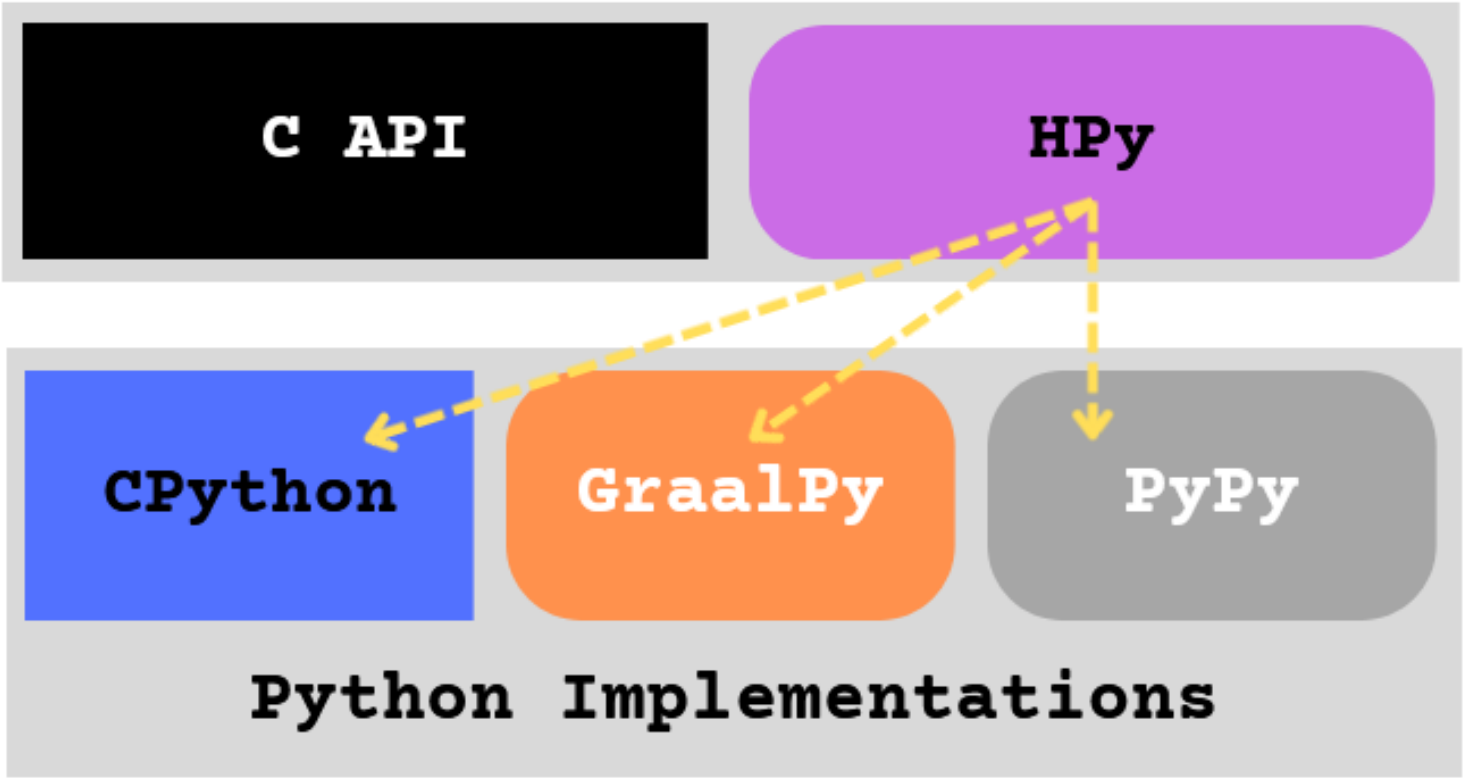
- ABI – API but for compiled code
- HPy libraries can be compiled in 3 different ways:
  - CPython ABI
  - Universal ABI
  - Hybrid ABI



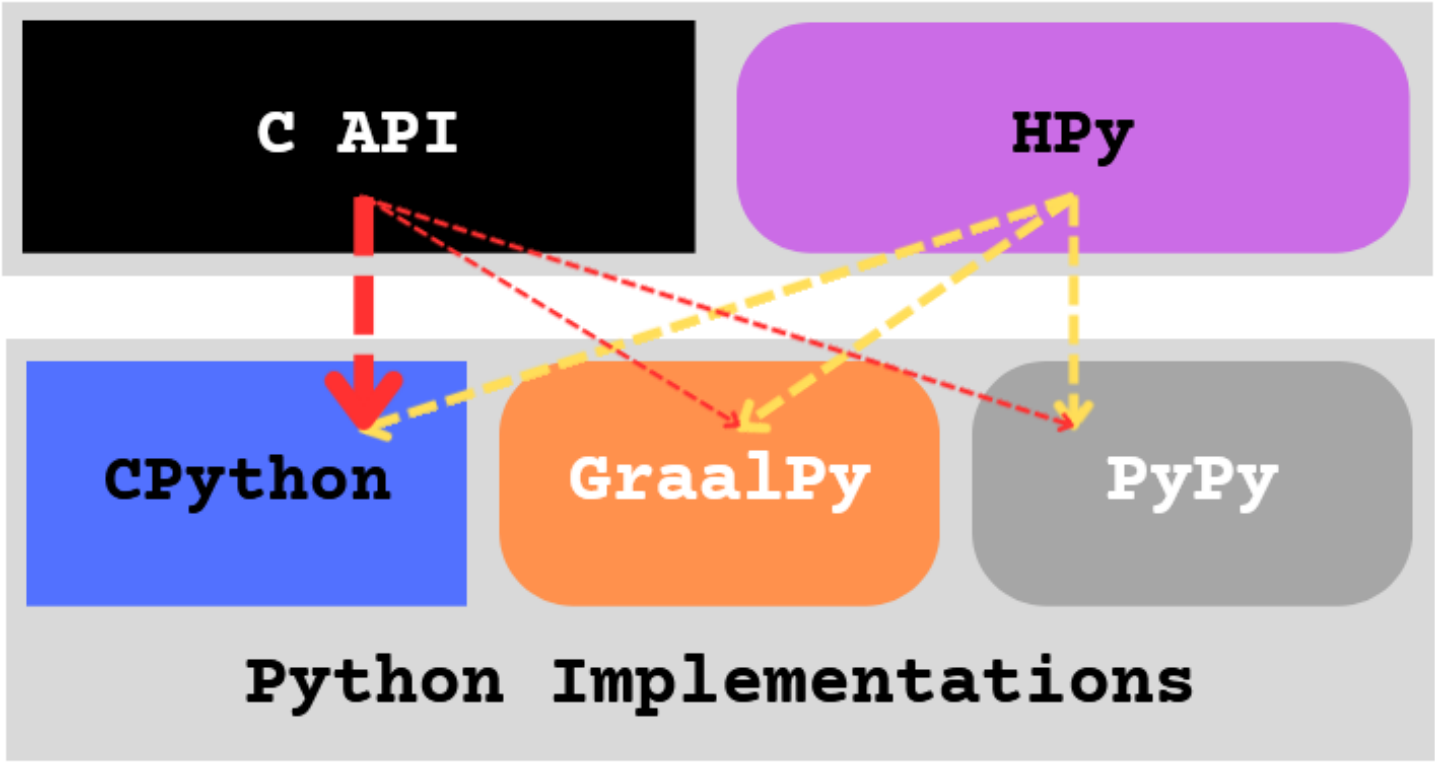
HPy – CPython ABI



HPy – Universal ABI

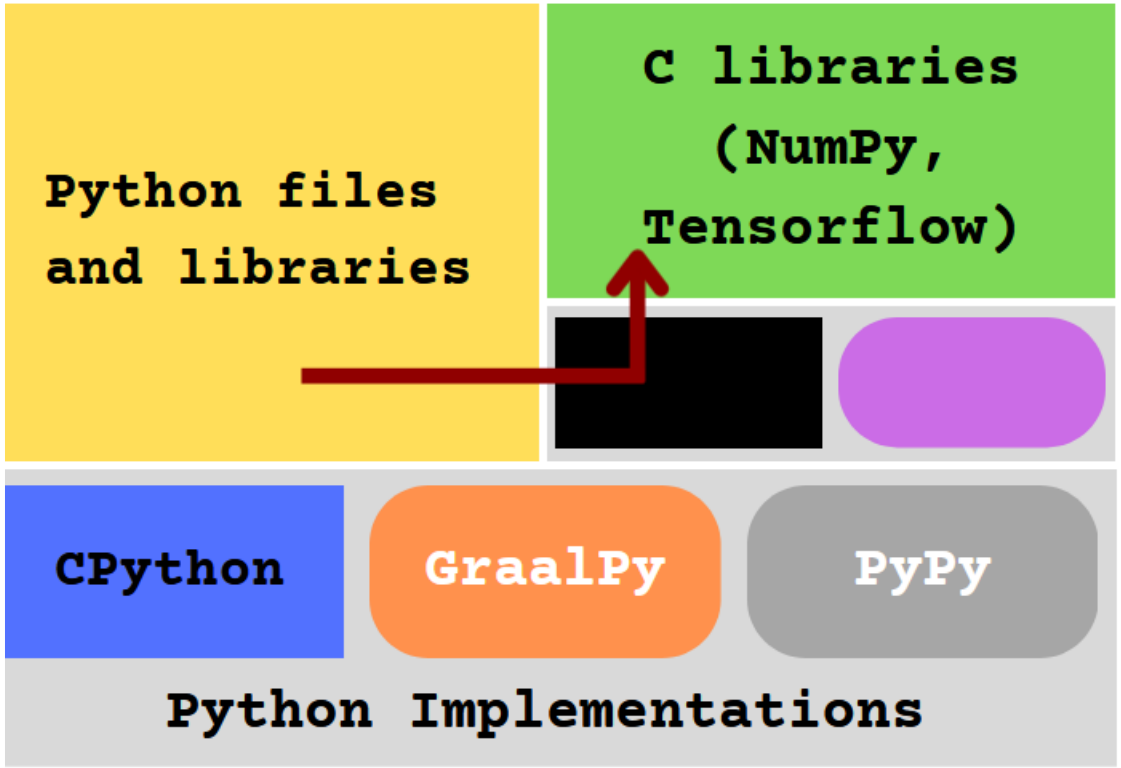


HPy – Hybrid ABI



Cython

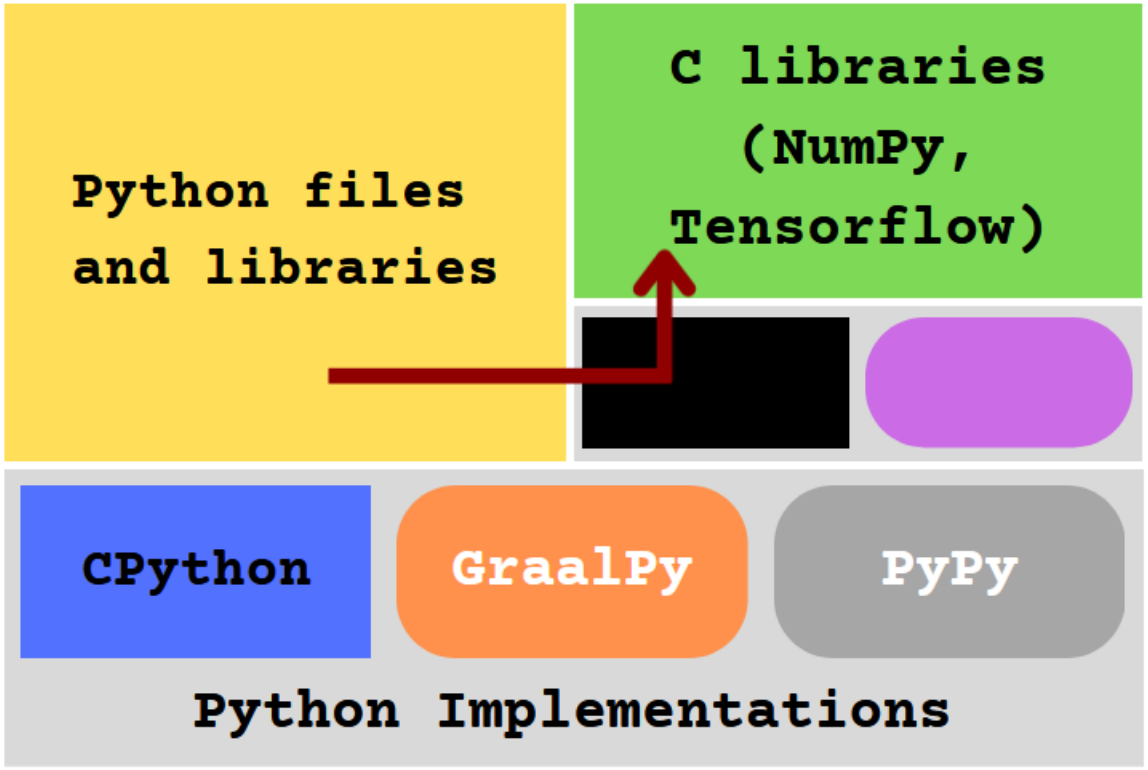
- Compiles Python code to C code
- C file can then compile to Python modules and be imported
- Can also compile `.pyx` files – Python files with some C features
- Can improve performance by more than 100 times



cython.org

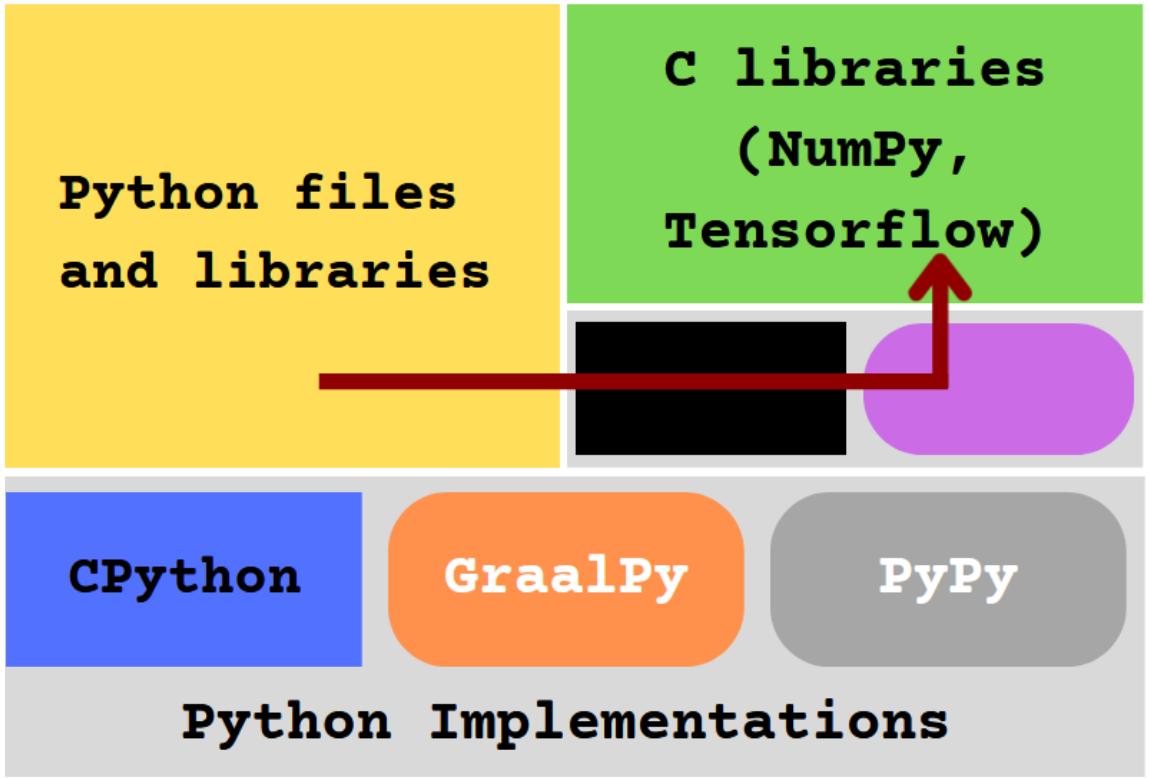
Cython

- The C API is used to create libraries from generated C files
- Libraries from Cython perform poorly on GraalPy
- Cython is incredibly optimised for CPython

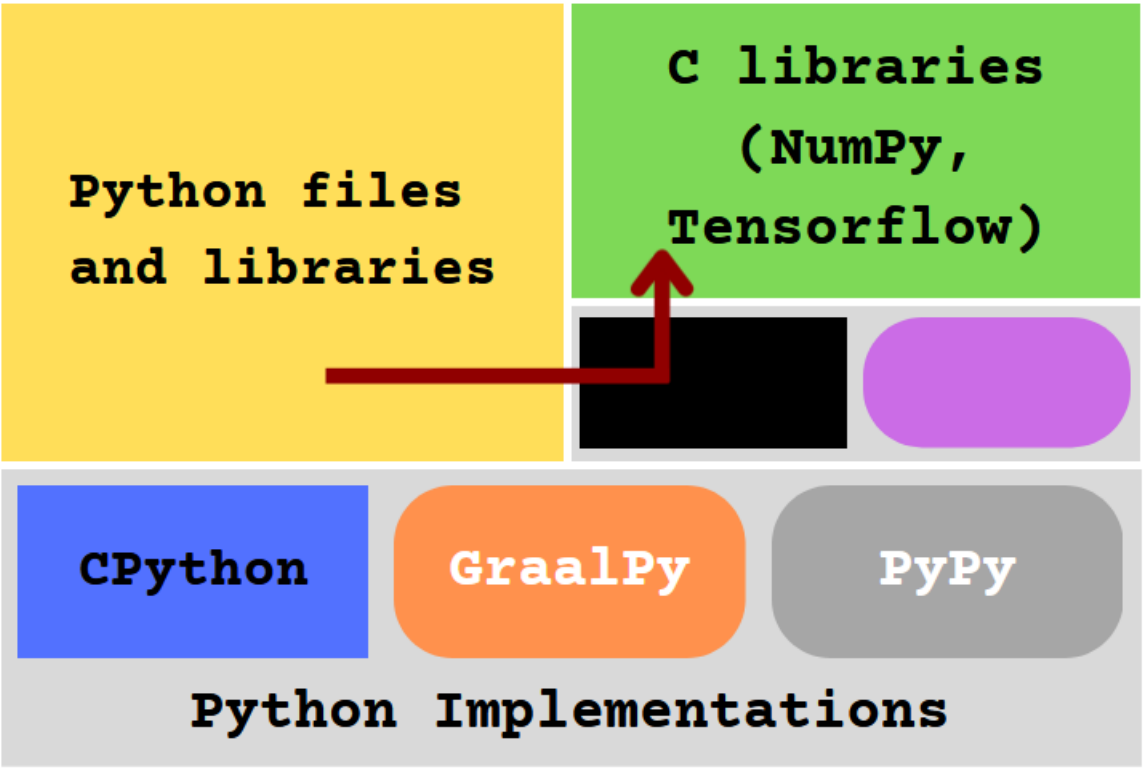


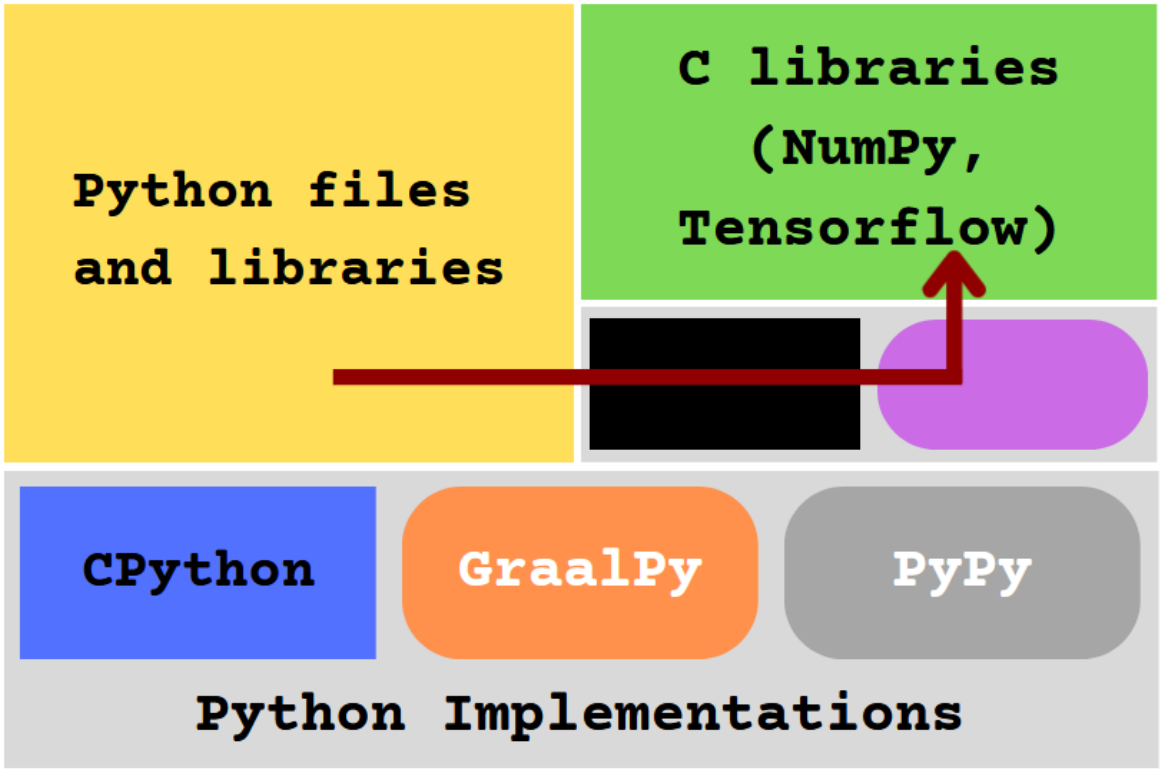
Cython with HPy

- This research focused on replacing the C API with HPy in C files generated by Cython
- This will allow Cython libraries to run much faster on GraalPy
- This will also enable the creation of HPy benchmarks from any Python file



Cython with HPy





Benchmarks

Forloop

single forloop that increments a number 20,000 times

Fibonacci

recursively calculates the 10th Fibonacci number

Float

performs sin, cos, and sqrt calculations on floats, artificial benchmark

Fannkuch

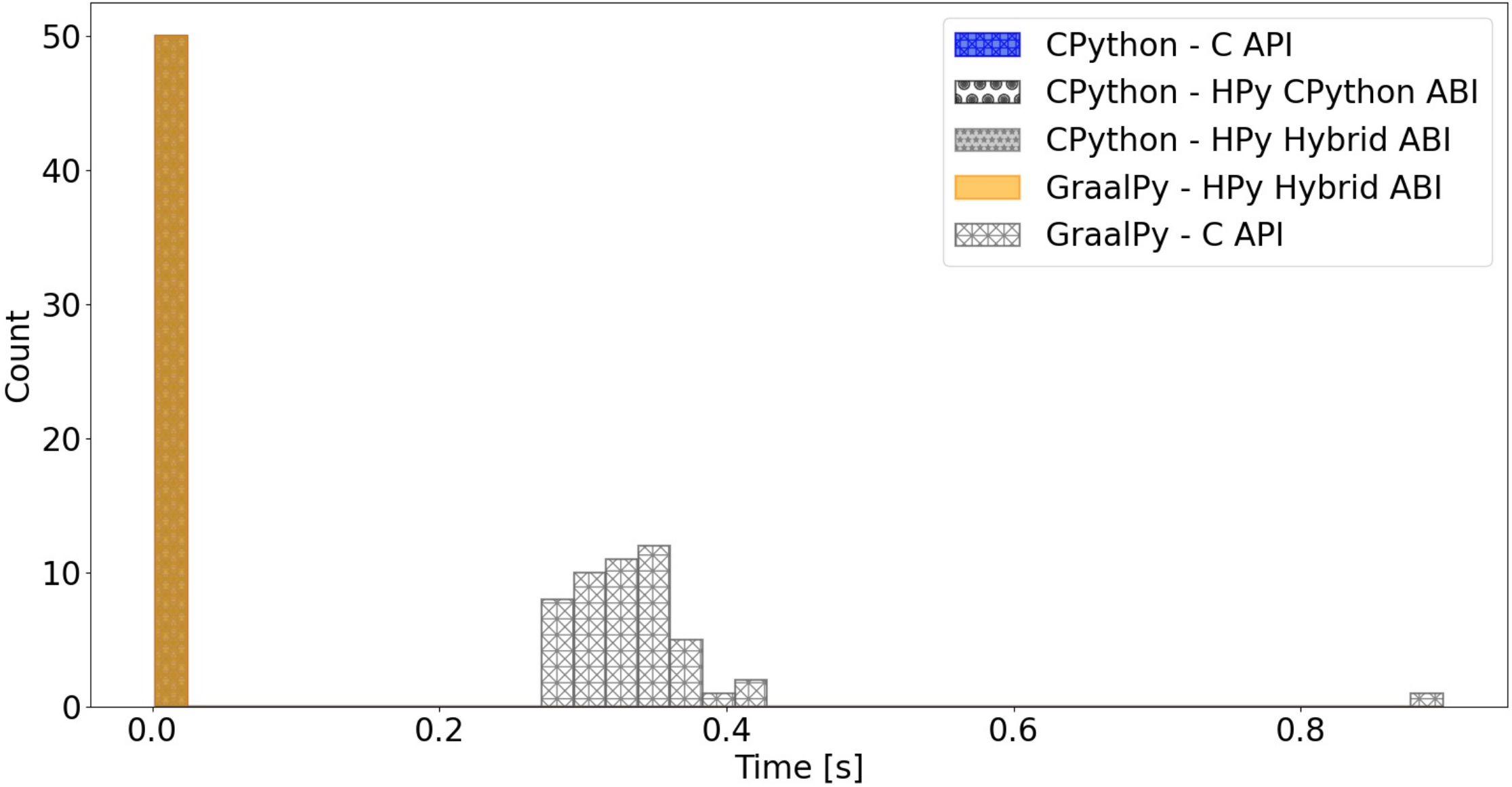
performs array flips and permutations to calculate minimums

Benchmarks

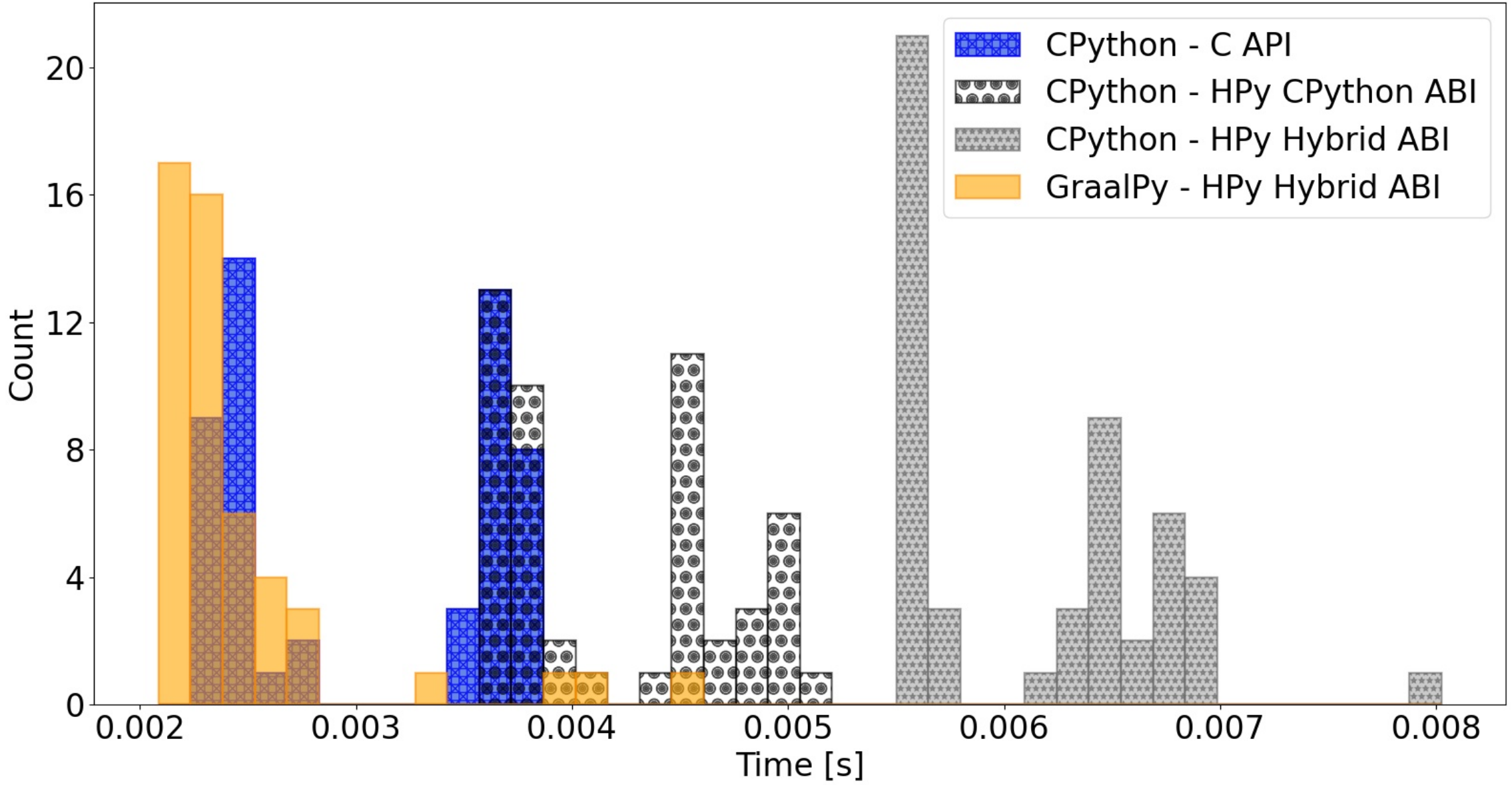
- 5 Benchmark Configurations
  1. **CPython** with **C API**
  2. **CPython** with **HPy CPython ABI**
  3. **CPython** with **HPy Hybrid ABI**
  4. **GraalPy** with **C API**
  5. **GraalPy** with **HPy Hybrid ABI**

Legend

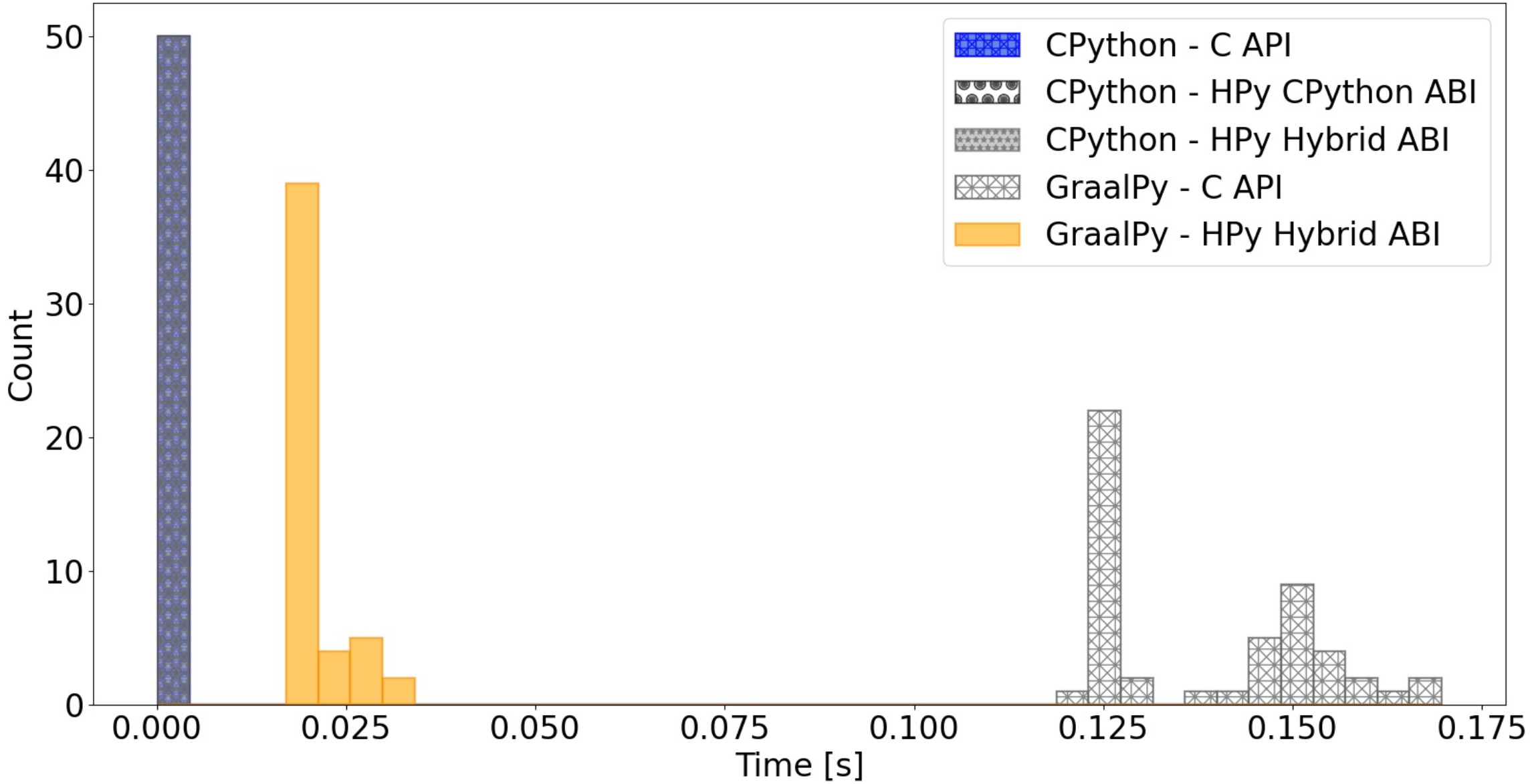
⬡Forloop


CPython - C API
CPython - HPy CPython ABI
CPython - HPy Hybrid ABI
GraalPy - HPy Hybrid ABI
GraalPy - C API
Count
0
10
20
30
40
50
0.0
0.2
0.4
0.6
0.8
Time [s]

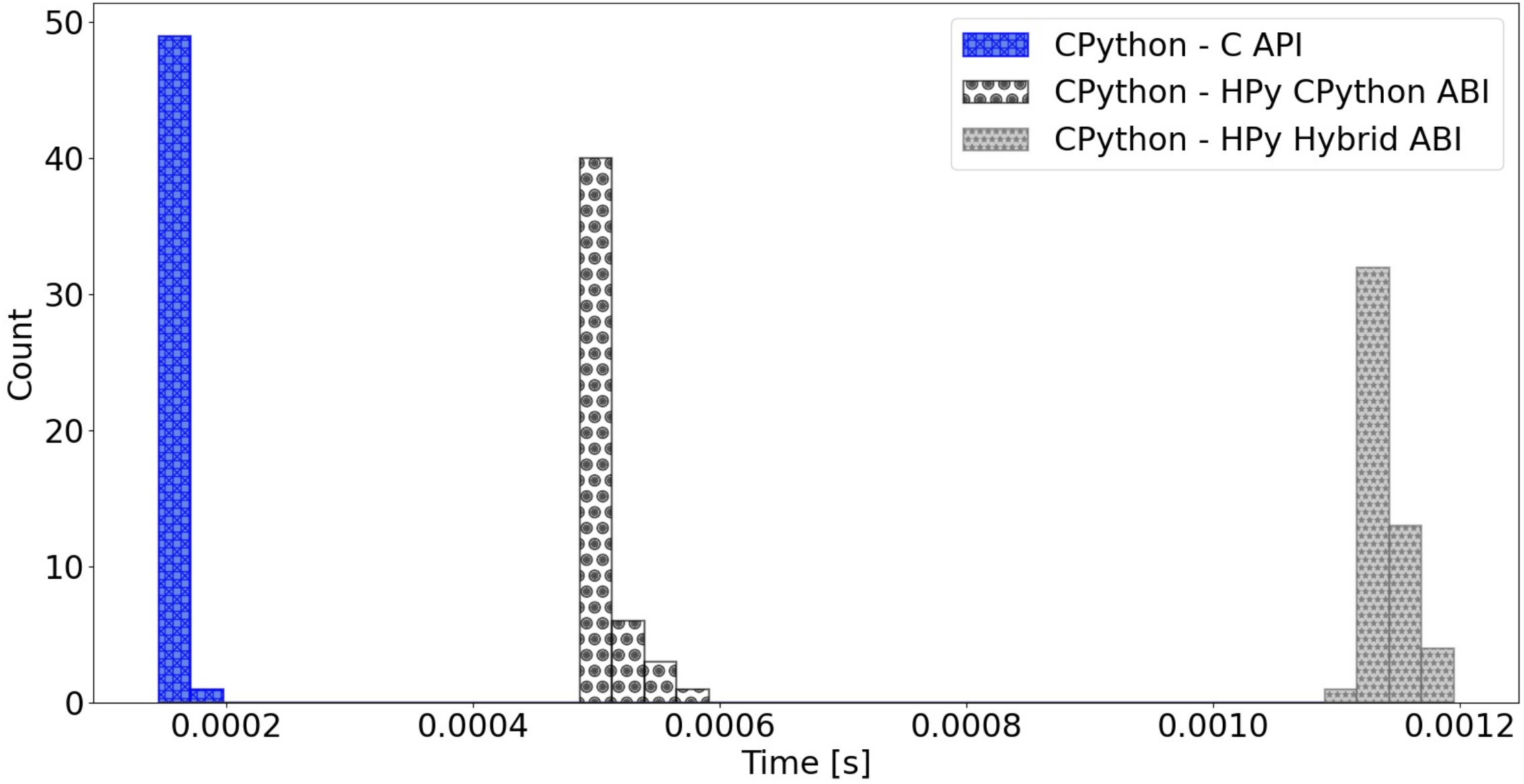
⬡Forloop


CPython - C API
CPython - HPy CPython ABI
CPython - HPy Hybrid ABI
GraalPy - HPy Hybrid ABI
Count
0
4
8
12
16
20
0.002
0.003
0.004
0.005
0.006
0.007
0.008
Time [s]

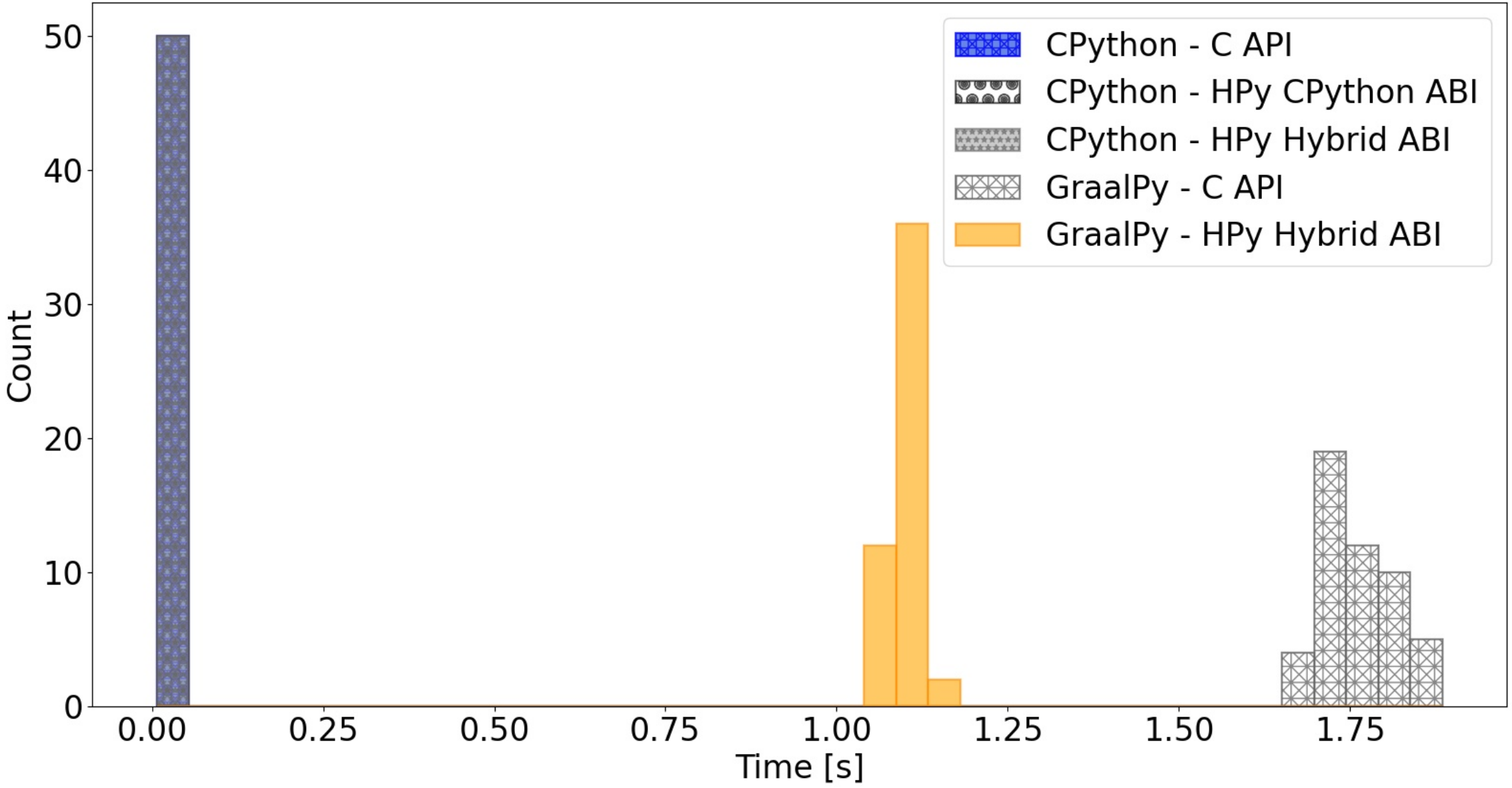
ΔFibonacci


CPython - C API
CPython - HPy CPython ABI
CPython - HPy Hybrid ABI
GraalPy - C API
GraalPy - HPy Hybrid ABI
Count
0
10
20
30
40
50
0.000
0.025
0.050
0.075
0.100
0.125
0.150
0.175
Time [s]

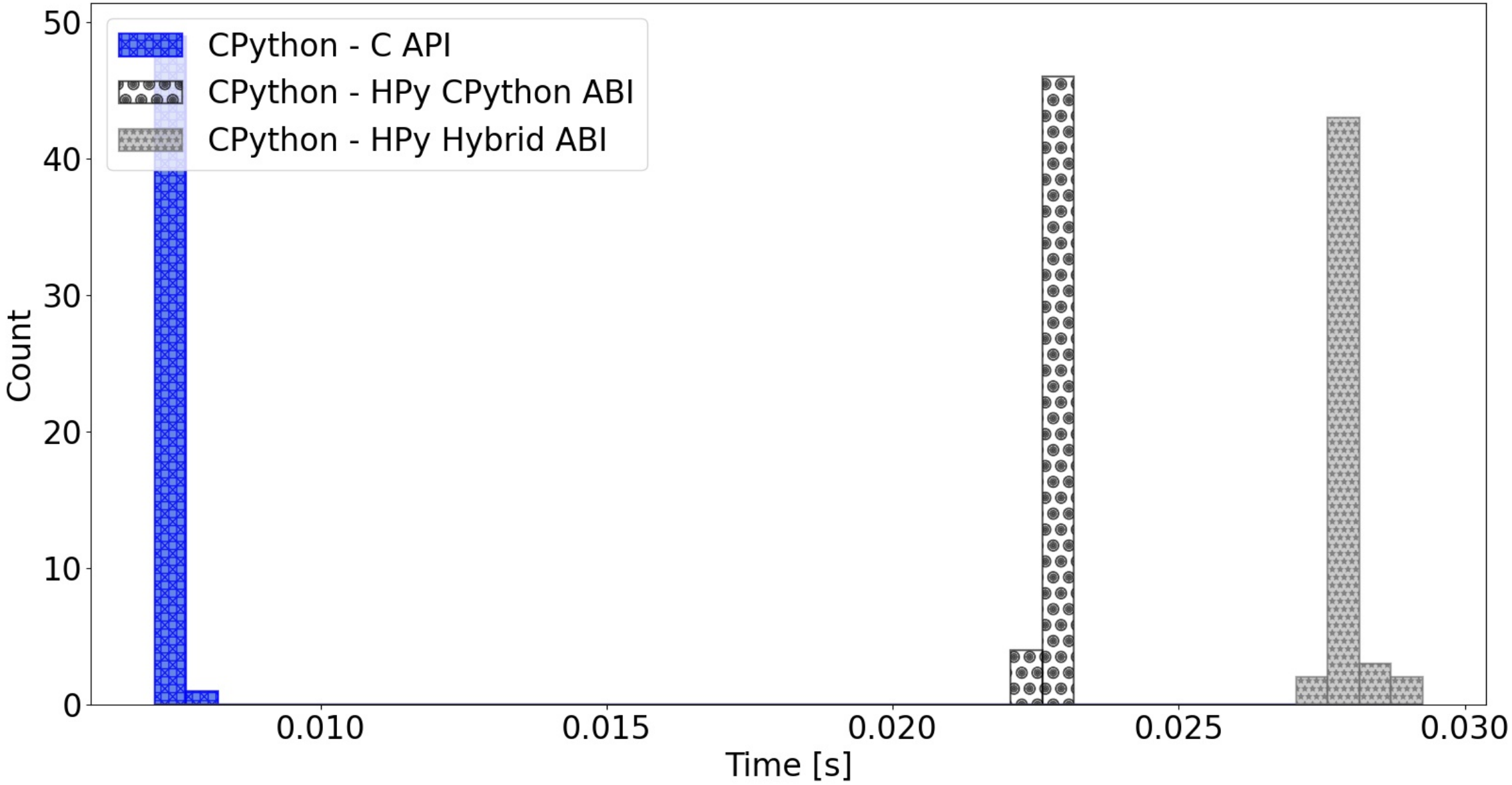
ΔFibonacci


CPython - C API
CPython - HPy CPython ABI
CPython - HPy Hybrid ABI
Count
0
10
20
30
40
50
0.0002
0.0004
0.0006
0.0008
0.0010
0.0012
Time [s]

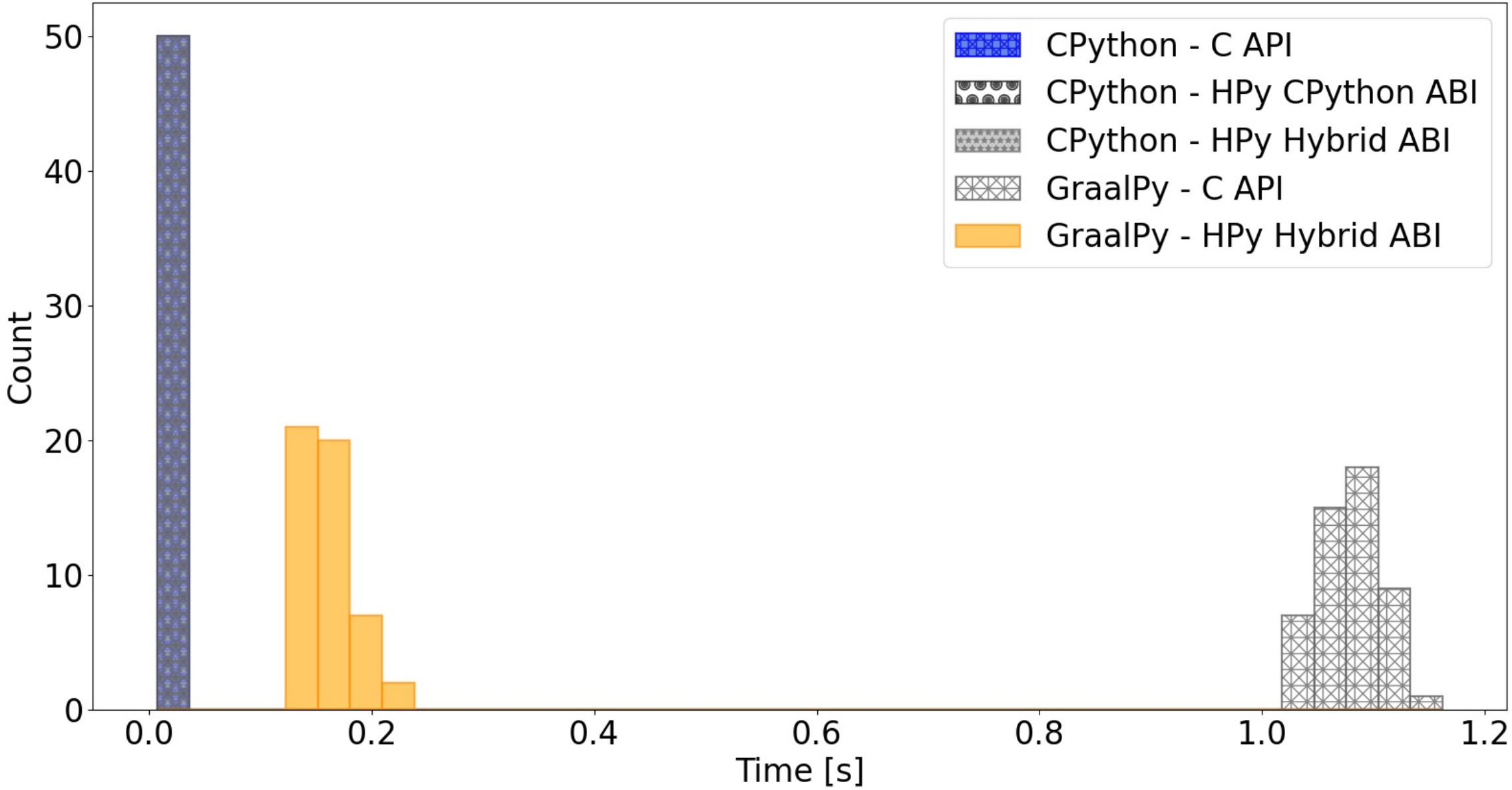
Float


CPython - C API
CPython - HPy CPython ABI
CPython - HPy Hybrid ABI
GraalPy - C API
GraalPy - HPy Hybrid ABI
Count
0
10
20
30
40
50
0.00
0.25
0.50
0.75
1.00
1.25
1.50
1.75
Time [s]

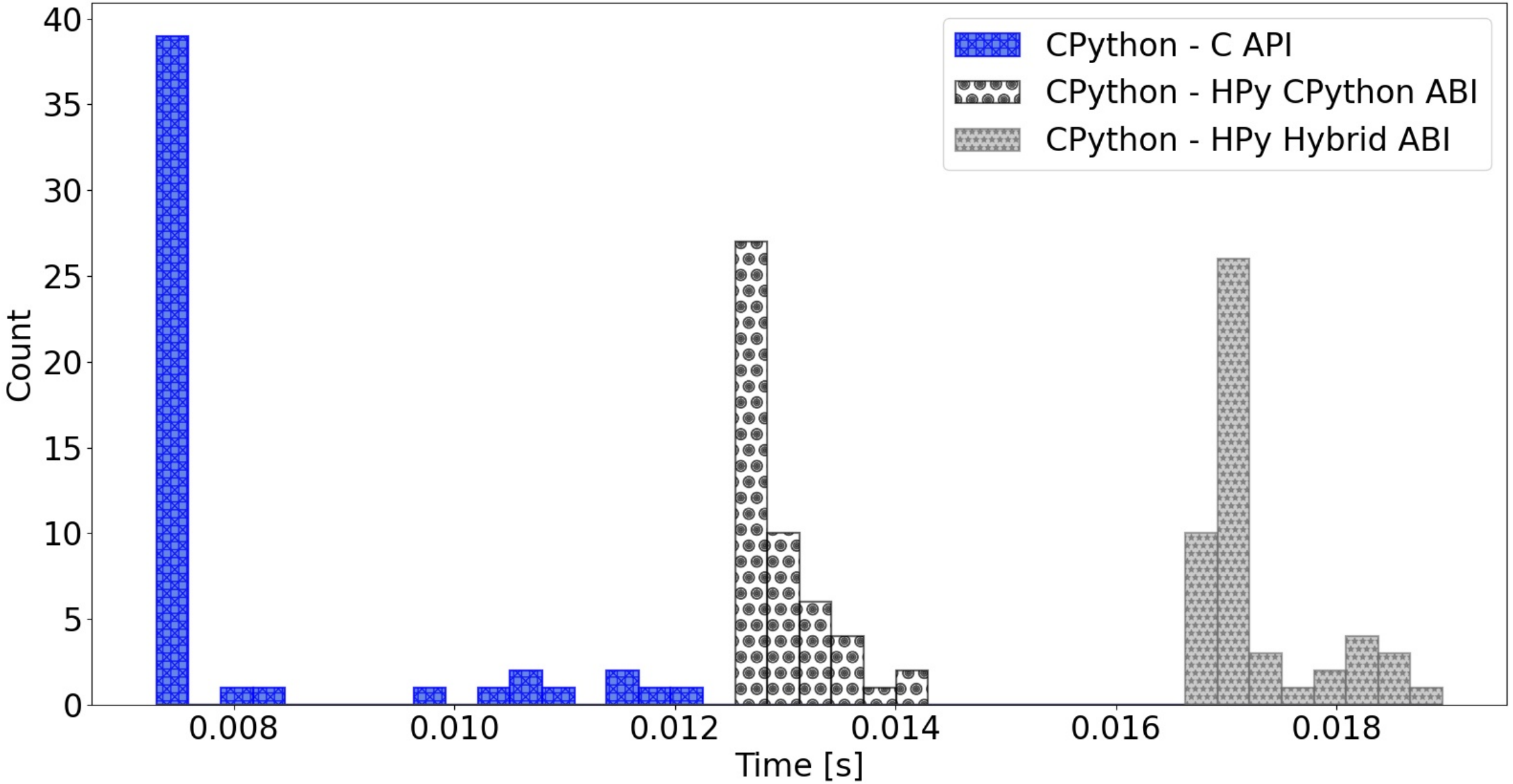
Float


CPython - C API
CPython - HPy CPython ABI
CPython - HPy Hybrid ABI
Count
0
10
20
30
40
50
0.010
0.015
0.020
0.025
0.030
Time [s]

Fannkuch


CPython - C API
CPython - HPy CPython ABI
CPython - HPy Hybrid ABI
GraalPy - C API
GraalPy - HPy Hybrid ABI
Count
0
10
20
30
40
50
0.0
0.2
0.4
0.6
0.8
1.0
1.2
Time [s]

Fannkuch


CPython - C API
CPython - HPy CPython ABI
CPython - HPy Hybrid ABI
Count
Time [s]
0
5
10
15
20
25
30
35
40
0.008
0.010
0.012
0.014
0.016
0.018

Conclusions

- **HPy** can improve Cython library performance on **GraalPy**
- **HPy** on **GraalPy** can equal the **C API** on **CPython**
- HPy does not meet performance expectations on **CPython**
- **HPy** is worth pursuing – Cython's **C API** optimisations are a special case
- There's still more work to do on the HPy backend for Cython

Acknowledgements

- Professor Michelle Kuttel – supervisor
- Oracle – Funding and Research Internship in Austria

- If you want to know more, see my work at https://github.com/DuToitSpies/cython/tree/hpy_backend

  or contact me at **dutoitspies@protonmail.com**

Conclusions

- **HPy** can improve Cython library performance on **GraalPy**
- **HPy** on **GraalPy** can equal the **C API** on **CPython**
- HPy does not meet performance expectations on **CPython**
- **HPy** is worth pursuing – Cython's **C API** optimisations are a special case
- There's still more work to do on the HPy backend for Cython

Conclusions

- **HPy** can improve Cython library performance on **GraalPy**
- **HPy** on **GraalPy** can equal the **C API** on **CPython**
- HPy does not meet performance expectations on **CPython**
- **HPy** is worth pursuing – Cython's **C API** optimisations are a special case
- There's still more work to do on the HPy backend for Cython