No time to idle about: Profiling import time in Python

Daniel Porteous

Production Engineer

dport.me/pycon.pdf

Me!

Me!!! Daniel Porteous!

Tweeter:

@banool1

Github:

github.com/banool

Website:

dport.me

Me!

Me!!! Daniel Porteous!

Tweeter:

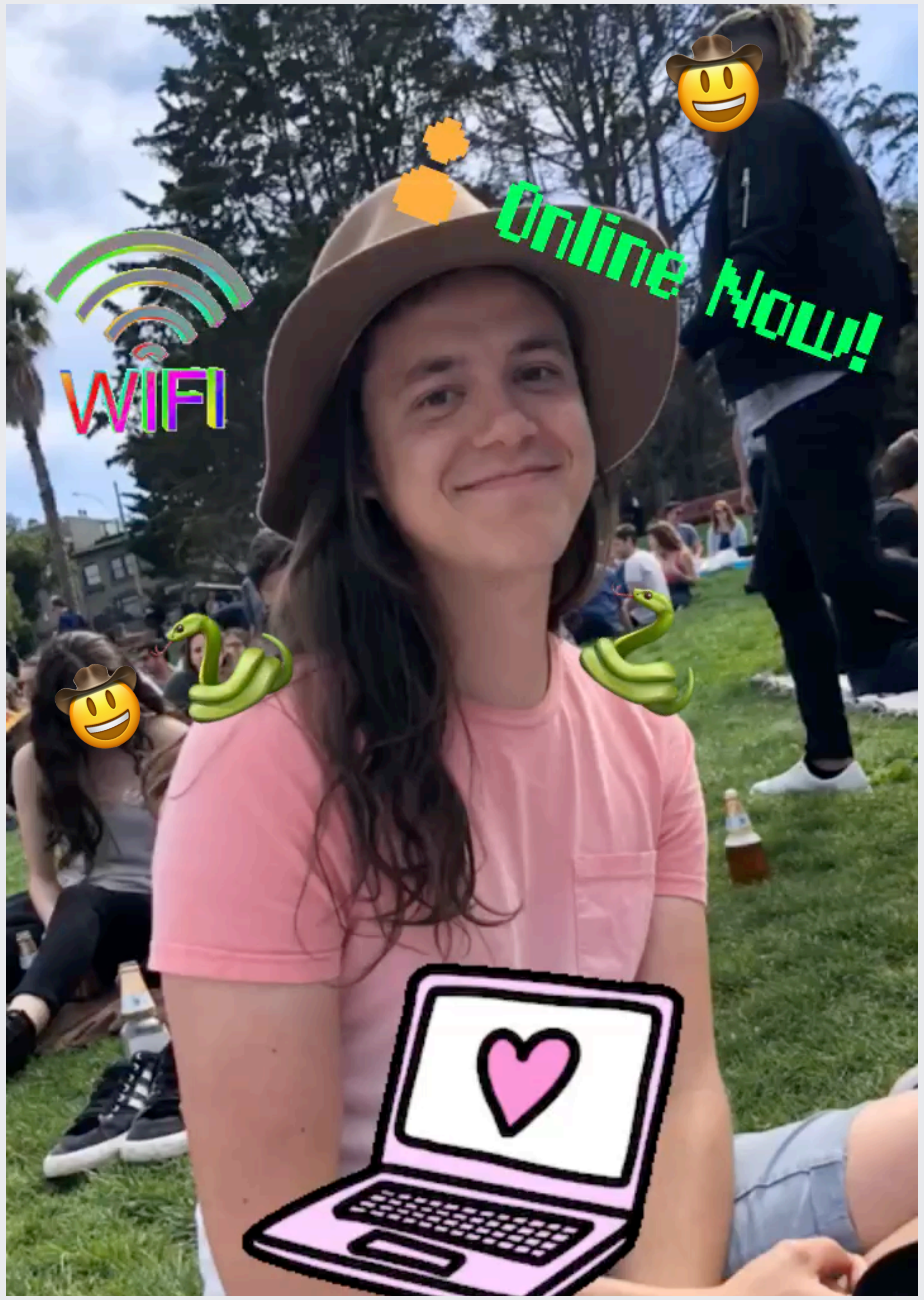
@banool1

Github:

github.com/banool

Website:

dport.me



The problem

Imports are slow, and you don’t know why

The problem

Imports are slow, and you don’t know why

- Why are slow imports a problem?

The problem

Imports are slow, and you don’t know why

- Why are slow imports a problem?
  - Mainly, command line tools.

The problem

Imports are slow, and you don’t know why

- Why are slow imports a problem?
  - Mainly, command line tools.
  - Worsens dev iteration time.

The problem

Imports are slow, and you don't know why

- Why are slow imports a problem?
  - Mainly, command line tools.
  - Worsens dev iteration time.
- How do we fix the problem?

The problem

Imports are slow, and you don't know why

- Why are slow imports a problem?
  - Mainly, command line tools.
  - Worsens dev iteration time.
- How do we fix the problem?
  - First, understand it.

dport.me/pycon.pdf

Looking at an example

Sloooooooooow imports 💤 💤 💤

dport.me/pycon.pdf

Looking at an example

Sloooooooooow imports zzZ zzZ zzZ

main.py

```python
import small
import large
print("Hi Pycon AU 2019!")
```

Looking at an example

Slooooooooooow imports 💤 💤 💤

main.py

```python
import small
import large
print("Hi Pycon AU 2019!")
```

small.py

```python
import time
time.sleep(1)
```

Looking at an example

Sloooooooooow imports 💤 💤 💤

main.py

```python
import small
import large
print("Hi Pycon AU 2019!")
```

small.py

```python
import time
time.sleep(1)
```

large.py

```python
import time
time.sleep(10)
```

Looking at an example

Slooooooooow imports 💤💤💤

main.py

```python
import small
import large
print("Hi Pycon AU 2019!")
```

small.py

```python
import time
time.sleep(1)
```

large.py

```python
import time
time.sleep(10)
```

```
$ time python main.py
Hi Pycon AU 2019!

real 0m11.061s
```

Looking at an example

Sloooooooooow imports 💤 💤 💤

main.py

```python
import small
import large
print("Hi Pycon AU 2019!")
```

small.py

```python
import time
time.sleep(1)
```

large.py

```python
import time
time.sleep(10)
```

```
$ time python main.py
Hi Pycon AU 2019!

real 0m11.061s
```

Per module import times

A few different approaches

Per module import times

A few different approaches

- Time each import directly in your own code.

Per module import times

A few different approaches

- Time each import directly in your own code.
  - Explicit time statements, context managers.

Per module import times

A few different approaches

- Time each import directly in your own code.
  - Explicit time statements, context managers.
- Modify module Finders / Loaders.

Per module import times

A few different approaches

- Time each import directly in your own code.
  - Explicit time statements, context managers.
- Modify module Finders / Loaders.
- Probes???

Per module import times

Enter 3.7

Per module import times

Enter 3.7

- `-X importtime` in 3.7.

Per module import times

Enter 3.7

- `-X importtime` in 3.7.
  - Also the `PYTHONPROFILEIMPORTTIME` environment variable.

Per module import times

Enter 3.7

- `-X importtime` in 3.7.
  - Also the `PYTHONPROFILEIMPORTTIME` environment variable.
- Supported directly in `Python/import.c`.

Per module import times

Enter 3.7

- `-X importtime` in 3.7.
  - Also the `PYTHONPROFILEIMPORTTIME` environment variable.
- Supported directly in `Python/import.c`.
- Thanks to the community and 3.7 contributors!

Per module import times

Enter 3.7

- `-X importtime` in 3.7.
  - Also the `PYTHONPROFILEIMPORTTIME` environment variable.
- Supported directly in `Python/import.c`.
- Thanks to the community and 3.7 contributors!
  - Special thanks to Victor Stinner and Inada Naoki!!!

dport.me/pycon.pdf

Back to our example

Still slow, who's the culprit!

main.py

```python
import small
import large
print("Hi!")
```

small.py

```python
import time
time.sleep(1)
```

large.py

```python
import time
time.sleep(10)
```

dport.me/pycon.pdf

Back to our example

Still slow, who's the culprit!

main.py

```python
import small
import large
print("Hi!")
```

small.py

```python
import time
time.sleep(1)
```

large.py

```python
import time
time.sleep(10)
```

```
$ python -X importtime main.py
```

Back to our example

Still slow, who's the culprit!

main.py

```python
import small
import large
print("Hi!")
```

small.py

```python
import time
time.sleep(1)
```

large.py

```python
import time
time.sleep(10)
```

```
$ python -X importtime main.py

$ PYTHONPROFILEIMPORTTIME="🤠"
  python main.py
```

Back to our example

Still slow, who’s the culprit!

main.py

```python
import small
import large
print("Hi!")
```

small.py

```python
import time
time.sleep(1)
```

large.py

```python
import time
time.sleep(10)
```

```
$ python -X importtime main.py
```

Back to our example

Still slow, who's the culprit!

main.py

```python
import small
import large
print("Hi!")
```

small.py

```python
import time
time.sleep(1)
```

large.py

```python
import time
time.sleep(10)
```

```
$ python -X importtime main.py
import time: self [us] | cumulative | imported package
import time:       145 |        145 | zipimport
import time:       737 |        737 | _frozen_importlib_external
import time:        73 |         73 |     _codecs
import time:      6401 |       6474 |   codecs
import time:      2292 |       2292 |   encodings.aliases
import time:      1780 |      10545 | encodings
import time:       512 |        512 | encodings.utf_8
import time:       115 |        115 | _signal
import time:       576 |        576 | encodings.latin_1
import time:        51 |         51 |     _abc
import time:      1233 |       1283 |   abc
import time:      1084 |       2367 | io
import time:        77 |         77 |       _stat
import time:      1144 |       1220 |     stat
import time:      1073 |       1073 |       genericpath
import time:      2533 |       3605 |     posixpath
import time:      4793 |       4793 |     _collections_abc
import time:      4456 |      14072 |   os
import time:      1277 |       1277 |   __builtin__
import time:       190 |        190 |     _locale
import time:       639 |        829 |   _bootlocale
import time:      1761 |       1761 |   sitecustomize
import time:      4901 |      22838 | site
import time:       719 |        719 |   time
import time:   1003709 |    1004428 | small
import time:  10003076 |   10003076 | large
Hi Pycon AU 2019!
```

Individual import times

Knowledge is power 🤔⏱️🐍

Individual import times

Knowledge is power 🤔⏱️🐍

```
$ python -X importtime main.py
import time: self [us] | cumulative | imported package
import time:       719 |        719 |   time
import time:   1003709 |    1004428 | small
import time:  10003076 |   10003076 | large
Hi Pycon AU 2019!
```

dport.me/pycon.pdf

```
$ python -X importtime main.py
import time: self [us] | cumulative | imported package
import time:       145 |        145 | zipimport
import time:       737 |        737 | _frozen_importlib_external
import time:        73 |         73 |     _codecs
import time:      6401 |       6474 |   codecs
import time:      2292 |       2292 |   encodings.aliases
import time:      1780 |      10545 | encodings
import time:       512 |        512 | encodings.utf_8
import time:       115 |        115 | _signal
import time:       576 |        576 | encodings.latin_1
import time:        51 |         51 |     _abc
import time:      1233 |       1283 |   abc
import time:      1084 |       2367 | io
import time:        77 |         77 |       _stat
import time:      1144 |       1220 |     stat
import time:      1073 |       1073 |       genericpath
import time:      2533 |       3605 |     posixpath
import time:      4793 |       4793 |     _collections_abc
import time:      4456 |      14072 |   os
import time:      1277 |       1277 |   __builtin__
import time:       190 |        190 |     _locale
import time:       639 |        829 |   _bootlocale
import time:      1761 |       1761 |   sitecustomize
import time:      4901 |      22838 | site
import time:       719 |        719 |   time
import time:   1003709 |    1004428 | small
import time:  10003076 |   10003076 | large
Hi Pycon AU 2019!
```

dport.me/pycon.pdf

```
$ python -X importtime main.py
import time: self [us] | cumulative | imported package
import time:       145 |        145 | zipimport
import time:       737 |        737 | _frozen_importlib_external
import time:        73 |         73 |     _codecs
import time:      6401 |       6474 |   codecs
import time:      2292 |       2292 |   encodings.aliases
import time:      1780 |      10545 | encodings
import time:       512 |        512 | encodings.utf_8
import time:       115 |        115 | _signal
import time:       576 |        576 | encodings.latin_1
import time:        51 |         51 |     _abc
import time:      1233 |       1283 |   abc
import time:      1084 |       2367 | io
import time:        77 |         77 |       _stat
import time:      1144 |       1220 |     stat
import time:      1073 |       1073 |       genericpath
import time:      2533 |       3605 |     posixpath
import time:      4793 |       4793 |     _collections_abc
import time:      4456 |      14072 |   os
import time:      1277 |       1277 |   __builtin__
import time:       190 |        190 |     _locale
import time:       639 |        829 |   _bootlocale
import time:      1761 |       1761 |   sitecustomize
import time:      4901 |      22838 | site
import time:       719 |        719 |   time
import time:   1003709 |    1004428 | small
import time:  10003076 |   10003076 | large
Hi Pycon AU 2019!
```

I’m not on 3.7 yet

Join the club 🤒 🤒 🤒

I’m not on 3.7 yet

Join the club 🤒 🤒 🤒

- import_times: https://github.com/banool/import_times

I’m not on 3.7 yet

Join the club 🤒 🤒 🤒

- import_times: https://github.com/banool/import_times
  - I made this one!

I’m not on 3.7 yet

Join the club 🤒 🤒 🤒

- import_times: https://github.com/banool/import_times
  - I made this one!

main.py

```python
import small
import large
print("Hi!")
```

I’m not on 3.7 yet

Join the club 🤒 🤒 🤒

- import_times: https://github.com/banool/import_times
  - I made this one!

main.py

```python
from import_times import enable_import_times
enable_import_times()

import small
import large
print("Hi Pycon AU 2019!")
```

I’m not on 3.7 yet

Join the club 🤒 🤒 🤒

- import_times: https://github.com/banool/import_times
  - I made this one!

```
$ python3.7 -X importtime main.py
import time: self [us] | cumulative | imported package
import time:       719 |        719 |   time
import time:   1003709 |    1004428 | small
import time:  10003076 |   10003076 | large
Hi Pycon AU 2019!
```

I’m not on 3.7 yet

Join the club 🤒 🤒 🤒

- import_times: https://github.com/banool/import_times
  - I made this one!

```
$ python3.6 main.py
import time: self [us] | cumulative | imported package
import time:   1005667 |    1005667 | small
import time:  10008714 |   10008714 | large
Hi Pycon AU 2019!
```

Visualising the data

🔎🔎🔎

Visualising the data
🔎🔎🔎

- Default output is great! 3.7 changed the game here.
  - The output still requires study.
- Visualisation is powerful because you can identify problems immediately.
- Flamegraphs are common visualisations of stack data.

Visualising the data

Structuring the data

Visualising the data

Structuring the data

```
$ python -X importtime main.py 2> main.stderr
Hi Pycon AU 2019!
```

Visualising the data

Structuring the data

```
$ python -X importtime main.py 2> main.stderr
Hi Pycon AU 2019!

$ cat main.stderr | python tree.py --basic
```

Visualising the data

Structuring the data

```
$ python -X importtime main.py 2> main.stderr
Hi Pycon AU 2019!

$ cat main.stderr | python tree.py --basic
everything 38523
 small 454
  time 528
 large 1496
```

Visualising the data

Structuring the data

```
$ python -X importtime main.py 2> main.stderr
Hi Pycon AU 2019!
```

Visualising the data

Structuring the data

```
$ python -X importtime main.py 2> main.stderr
Hi Pycon AU 2019!

$ cat main.stderr | python tree.py --flame > main.tree
```

Visualising the data

Structuring the data

```
$ python -X importtime main.py 2> main.stderr
Hi Pycon AU 2019!

$ cat main.stderr | python tree.py --flame > main.tree

$ tail -n 3 main.tree
small 454
small;time 528
large 1496
```

Visualising the data

Structuring the data

```
$ python -X importtime main.py 2> main.stderr
Hi Pycon AU 2019!

$ cat main.stderr | python tree.py --flame > main.tree

$ tail -n 3 main.tree
small 454
small;time 528
large 1496

$ flamegraph.pl main.tree --flamechart tree > main.svg
```

Visualising the data

Structuring the data

Flame Chart
Search
p..
_collections..
c..
e..
os
sitec..
time
_..
encodings
io
site
small
large

Visualising the data

🐟 🐟 🐟

Visualising the data

🐟 🐟 🐟

- Tuna: https://github.com/nschloe/tuna

Visualising the data

🐟 🐟 🐟

- Tuna: https://github.com/nschloe/tuna
  - Big thanks to Nico Schloemer!

Visualising the data

🐟 🐟 🐟

- Tuna: https://github.com/nschloe/tuna
  - Big thanks to Nico Schloemer!

```
$ python -X importtime main.py &> main.stderr
```

Visualising the data

🐟 🐟 🐟

- Tuna: https://github.com/nschloe/tuna
  - Big thanks to Nico Schloemer!

```
$ python -X importtime main.py &> main.stderr

$ tuna main.stderr
# Starts up a Python webserver
```

dport.me/pycon.pdf

Visualising the data

Structuring the data

tuna
main
0.029 s (100.0%)
large
0.011 s (39.3%)
site
0.007 s (23.6%)
small
0.004 s (14.7%)
encodings
0.003 s (10.8%)
io
os
0.003 s (11.9%)
sitecustomize
0.002 s (6.2%)
time
abc
ollections_a
.002 s (5.3%

Visualising the data

Is a tree 🌳 the right data structure?

Visualising the data

Is a tree 🌳 the right data structure?

The import structure isn’t strictly a tree, it’s a graph

Visualising the data

Is a tree 🌳 the right data structure?

The import structure isn’t strictly a tree, it’s a graph

```python
sys.modules = []

def import(module):
    for m in module.imports:
        if m in sys.modules:
            continue
        import(m)
        sys.modules.append(m)
```

Visualising the data

Demonstrating import behaviour

dport.me/pycon.pdf

Visualising the data

Demonstrating import behaviour

main.py

```python
import small
import large
print("Hi!")
```

small.py

```python
import time
time.sleep(1)
```

large.py

```python
import time
time.sleep(10)
```

Visualising the data

Demonstrating import behaviour

main.py

```python
import small
import large
print("Hi!")
```

small.py

```python
import time
time.sleep(1)
```

large.py

```python
import time
time.sleep(10)
```

Visualising the data

Demonstrating import behaviour

main.py

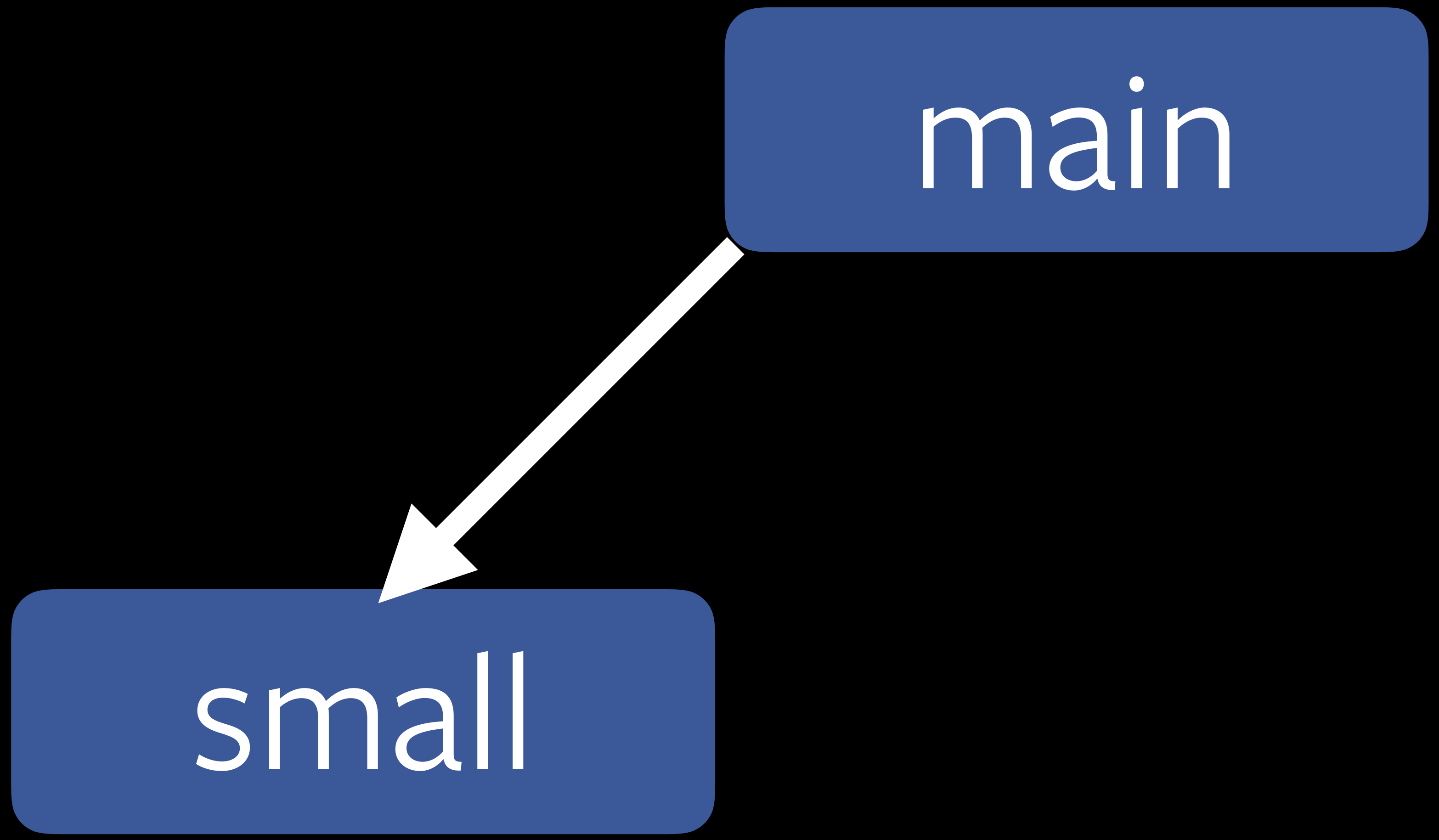
```python
import small
import large
print("Hi!")
```

small.py

```python
import time
time.sleep(1)
```

large.py

```python
import time
time.sleep(10)
```



dport.me/pycon.pdf

Visualising the data

Demonstrating import behaviour

main.py

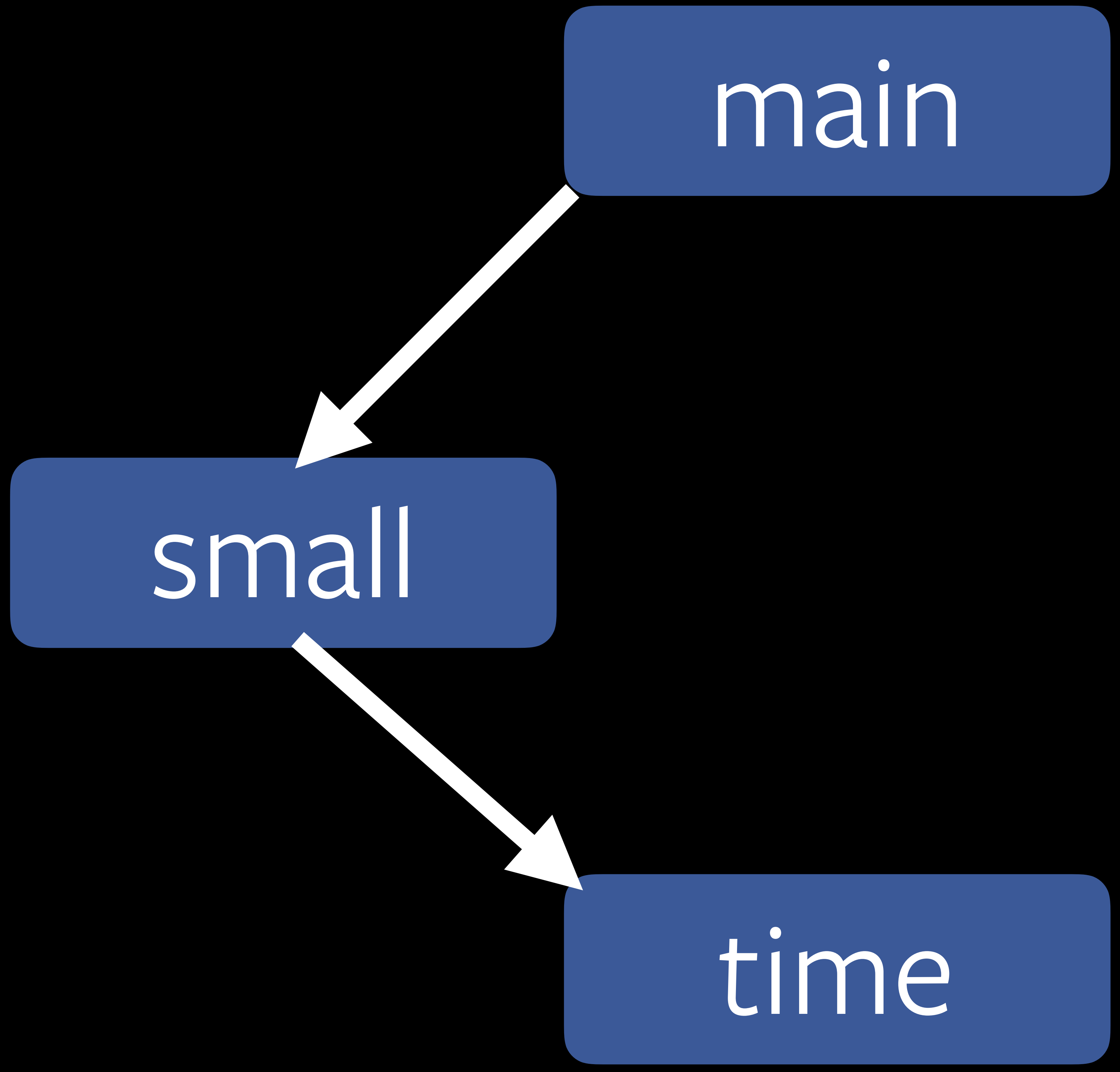
```python
import small
import large
print("Hi!")
```

small.py

```python
import time
time.sleep(1)
```

large.py

```python
import time
time.sleep(10)
```



Visualising the data

Demonstrating import behaviour

main.py

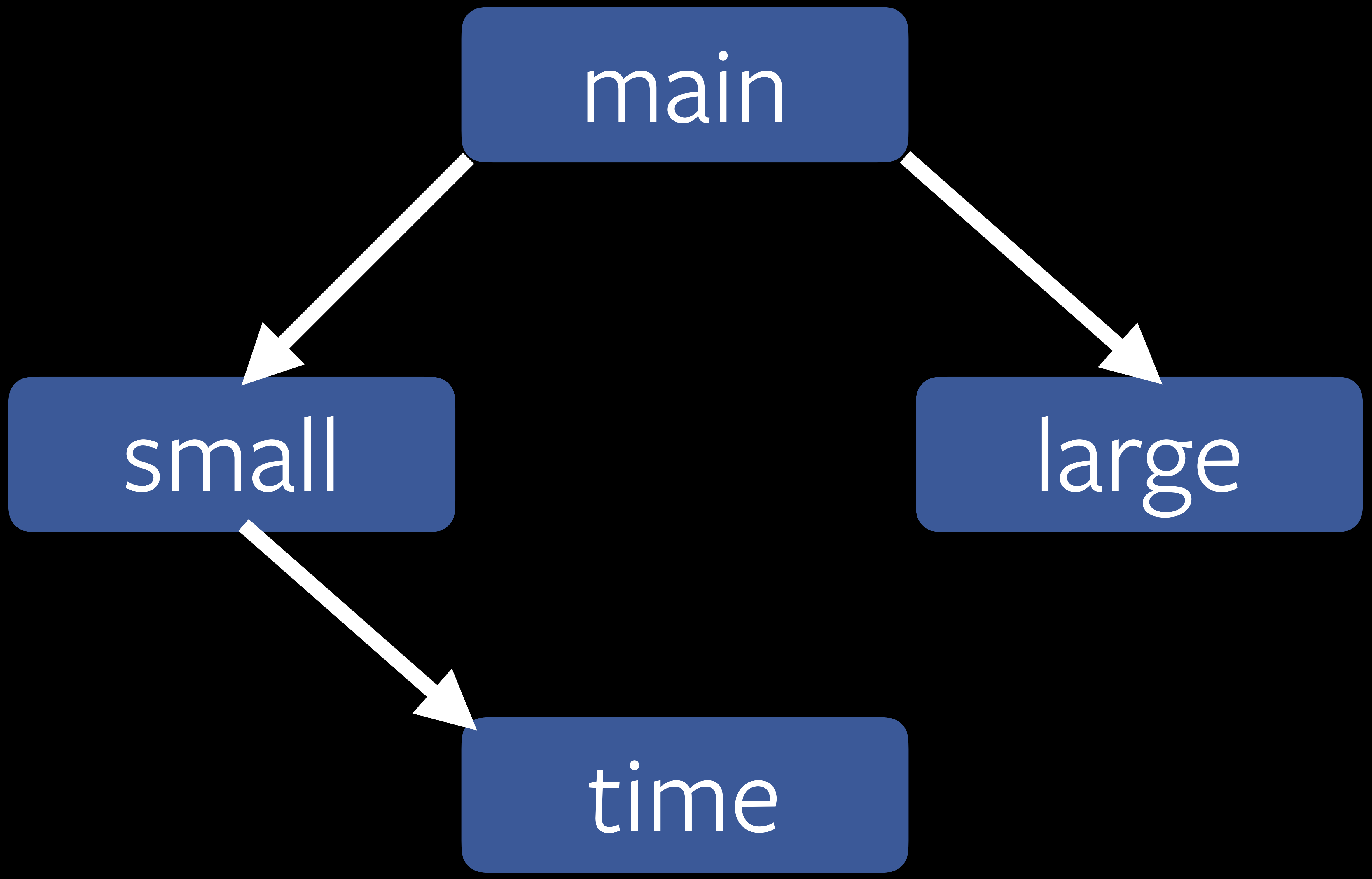
```python
import small
import large
print("Hi!")
```

small.py

```python
import time
time.sleep(1)
```

large.py

```python
import time
time.sleep(10)
```


$$\begin{tikzcd}
 & \text{main} \arrow[dl] \arrow[dr] & \\
\text{small} \arrow[dr] & & \text{large} \\
 & \text{time} &
\end{tikzcd}$$

Visualising the data

Demonstrating import behaviour

main.py

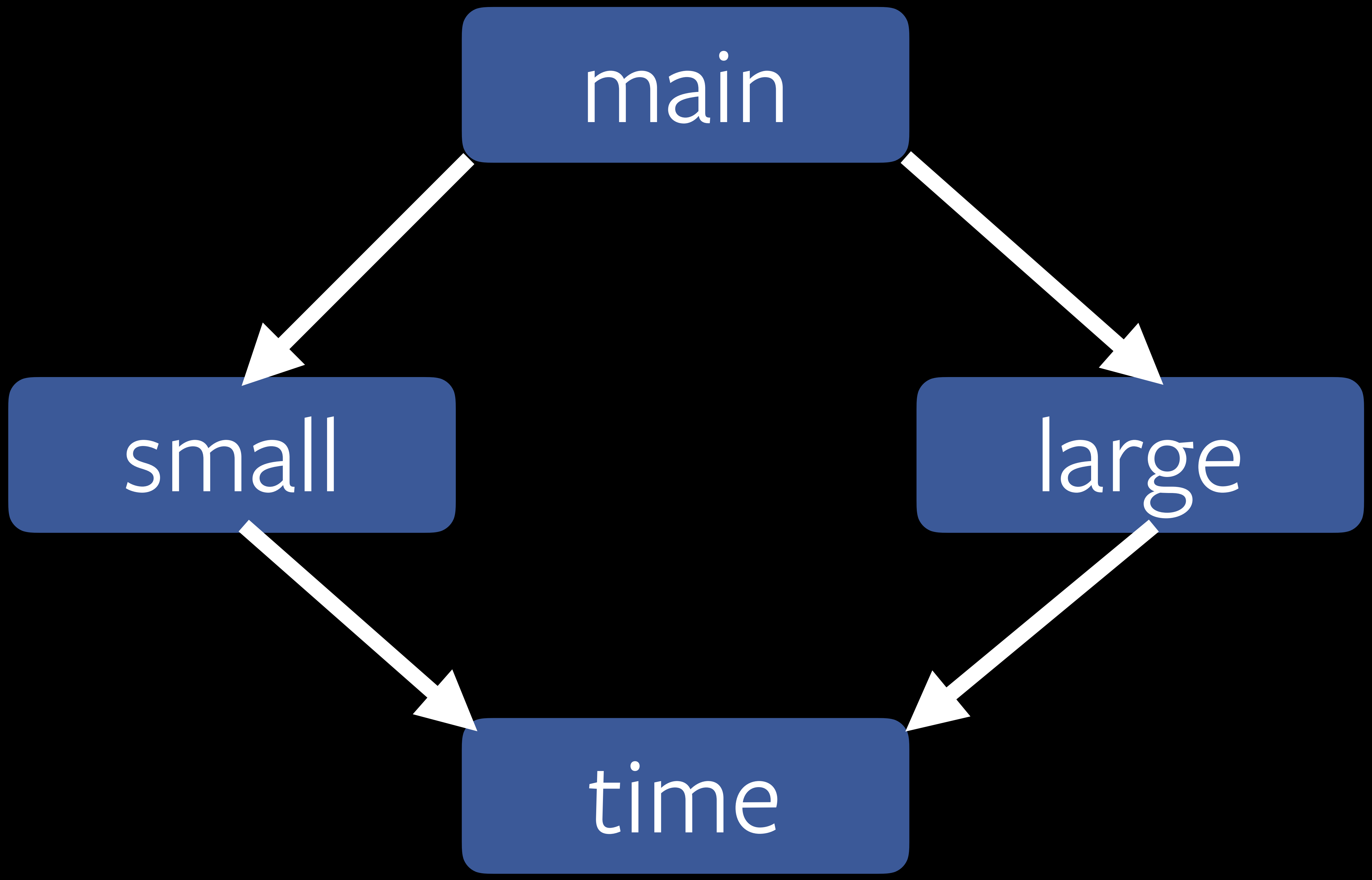
```python
import small
import large
print("Hi!")
```

small.py

```python
import time
time.sleep(1)
```

large.py

```python
import time
time.sleep(10)
```


$$\begin{tikzcd}
 & \text{main} \arrow[dl] \arrow[dr] & \\
\text{small} \arrow[dr] & & \text{large} \arrow[dl] \\
 & \text{time} &
\end{tikzcd}$$

Visualising the data

Demonstrating import behaviour

main.py

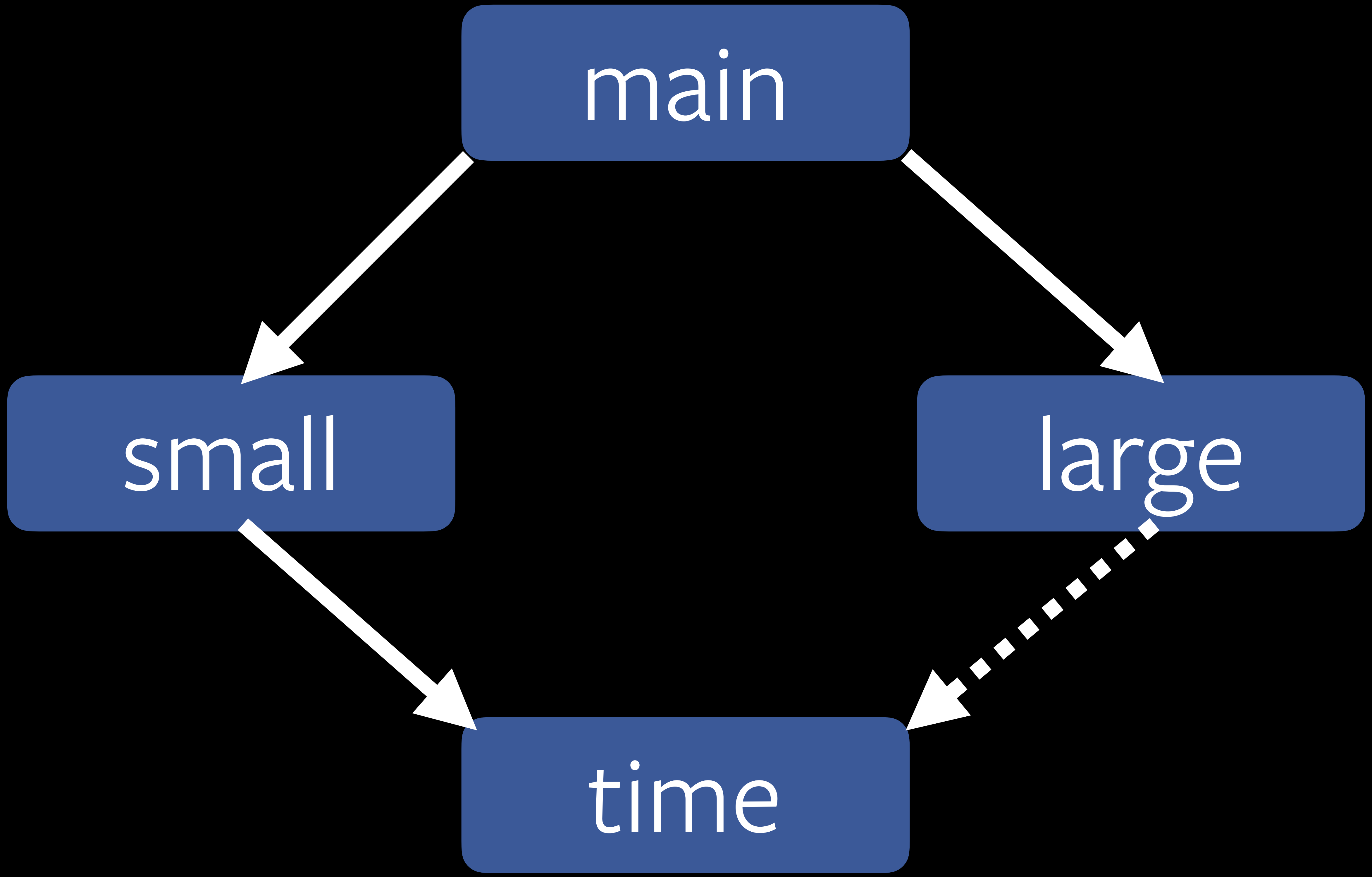
```python
import small
import large
print("Hi!")
```

small.py

```python
import time
time.sleep(1)
```

large.py

```python
import time
time.sleep(10)
```


$$\begin{tikzcd} & \text{main} \arrow[dl] \arrow[dr] & \\ \text{small} \arrow[dr] & & \text{large} \arrow[dl, dotted] \\ & \text{time} & \end{tikzcd}$$

Visualising the data

Demonstrating import behaviour

main.py

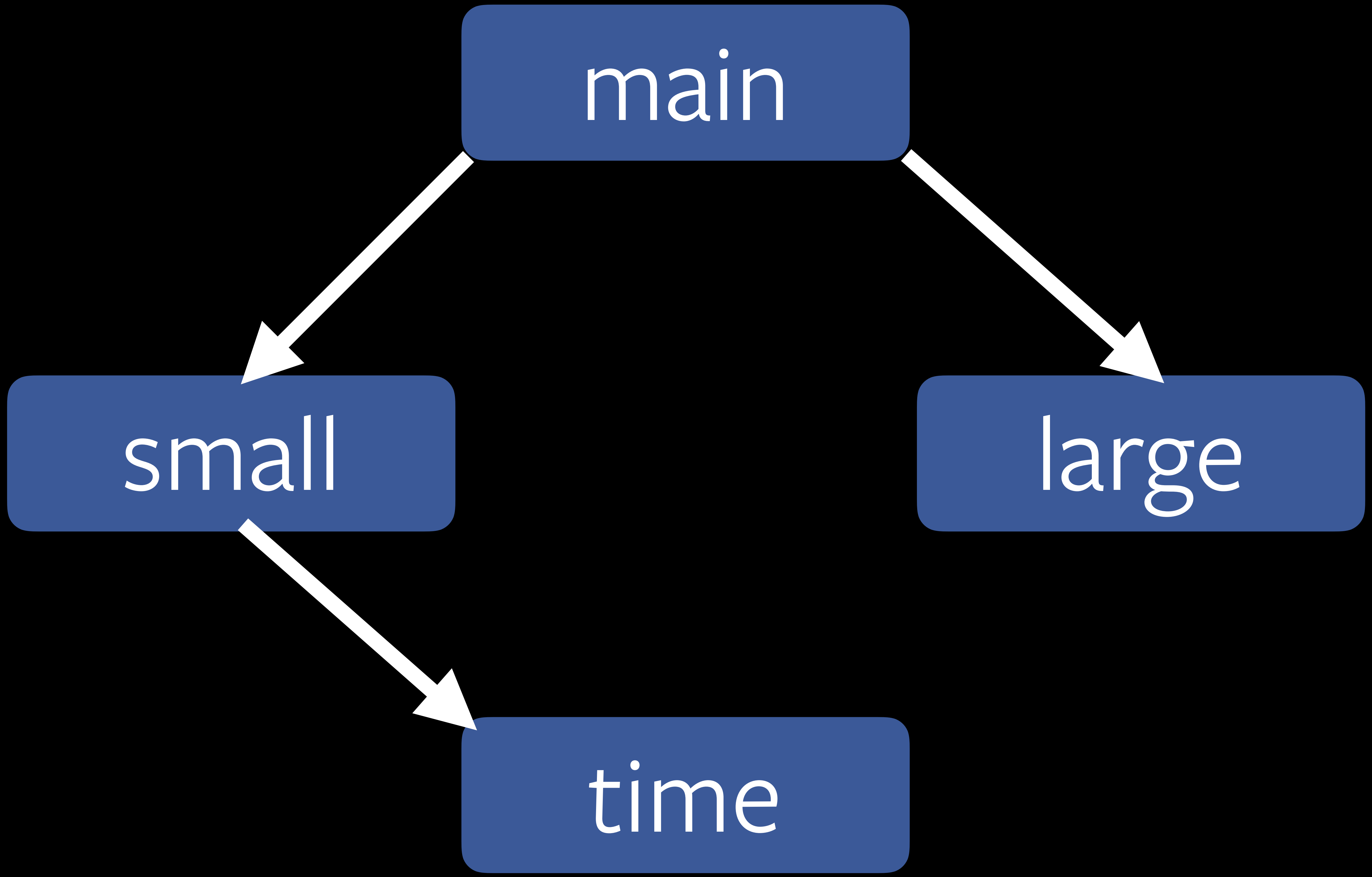
```python
import small
import large
print("Hi!")
```

small.py

```python
import time
time.sleep(1)
```

large.py

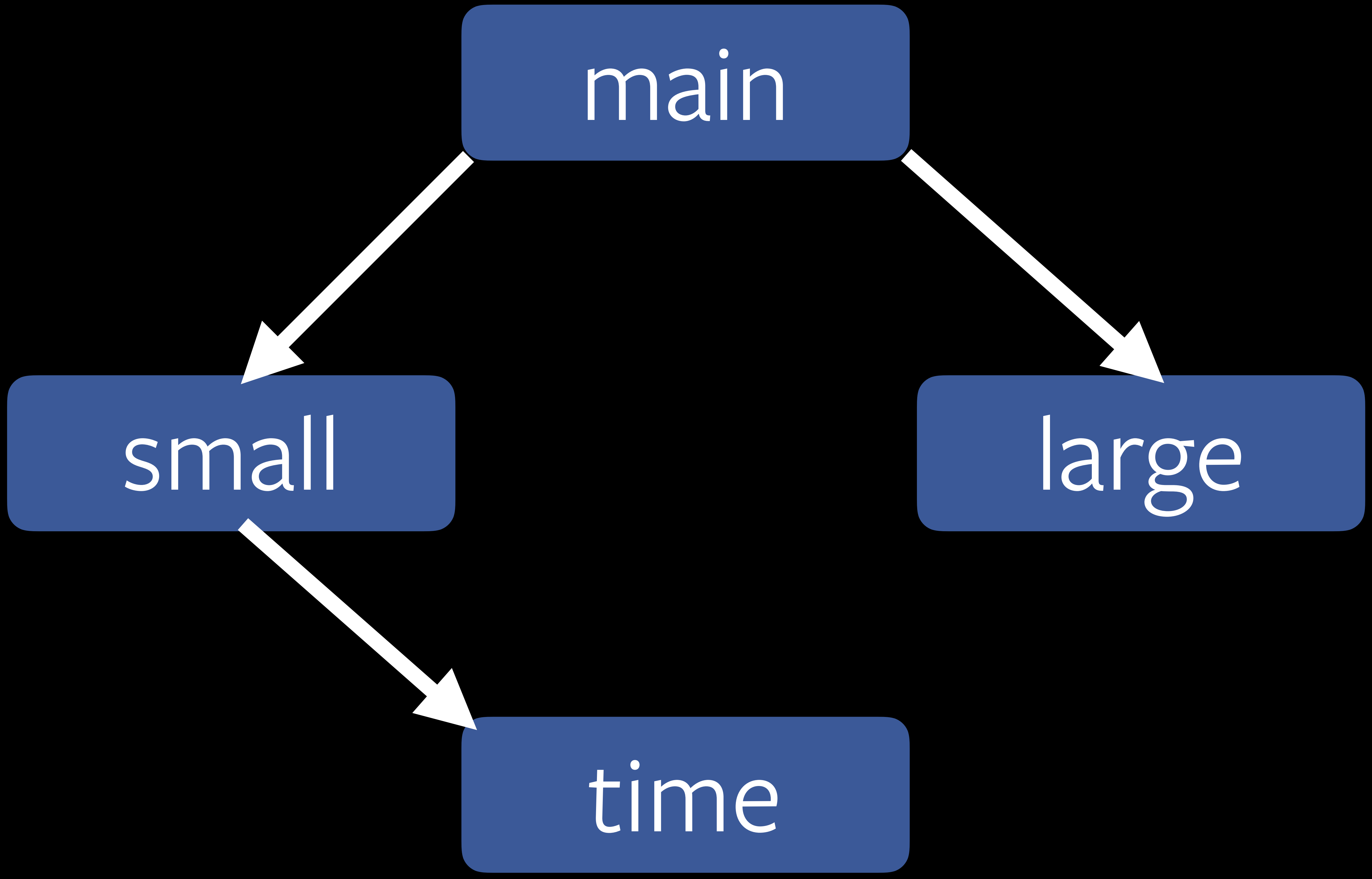
```python
import time
time.sleep(10)
```



Visualising the data

Demonstrating import behaviour

```
self   | cumulative | package
  528 |        528 |   time
  454 |        981 | small
 1496 |       1496 | large
```



Visualising the data

Showing imports as a graph

Visualising the data

Showing imports as a graph

- Pydeps: https://github.com/thebjorn/pydeps

Visualising the data

Showing imports as a graph

- Pydeps: https://github.com/thebjorn/pydeps
  - Open source project that produces import graphs

Visualising the data

Showing imports as a graph

- Pydeps: https://github.com/thebjorn/pydeps
  - Open source project that produces import graphs
  - Big thanks to Bjørn Pettersen!

Visualising the data

Showing imports as a graph

- Pydeps: https://github.com/thebjorn/pydeps
  - Open source project that produces import graphs
  - Big thanks to Bjørn Pettersen!

```
$ pydeps main.py --max-bacon 0 -o main.svg
```

Visualising the data

Showing imports as a graph

Visualising the data

Showing imports as a graph

main.py

```python
import small
import large
print("Hi!")
```

small.py

```python
import time
time.sleep(1)
```

large.py

```python
import time
time.sleep(10)
```

Visualising the data

Showing imports as a graph

main.py

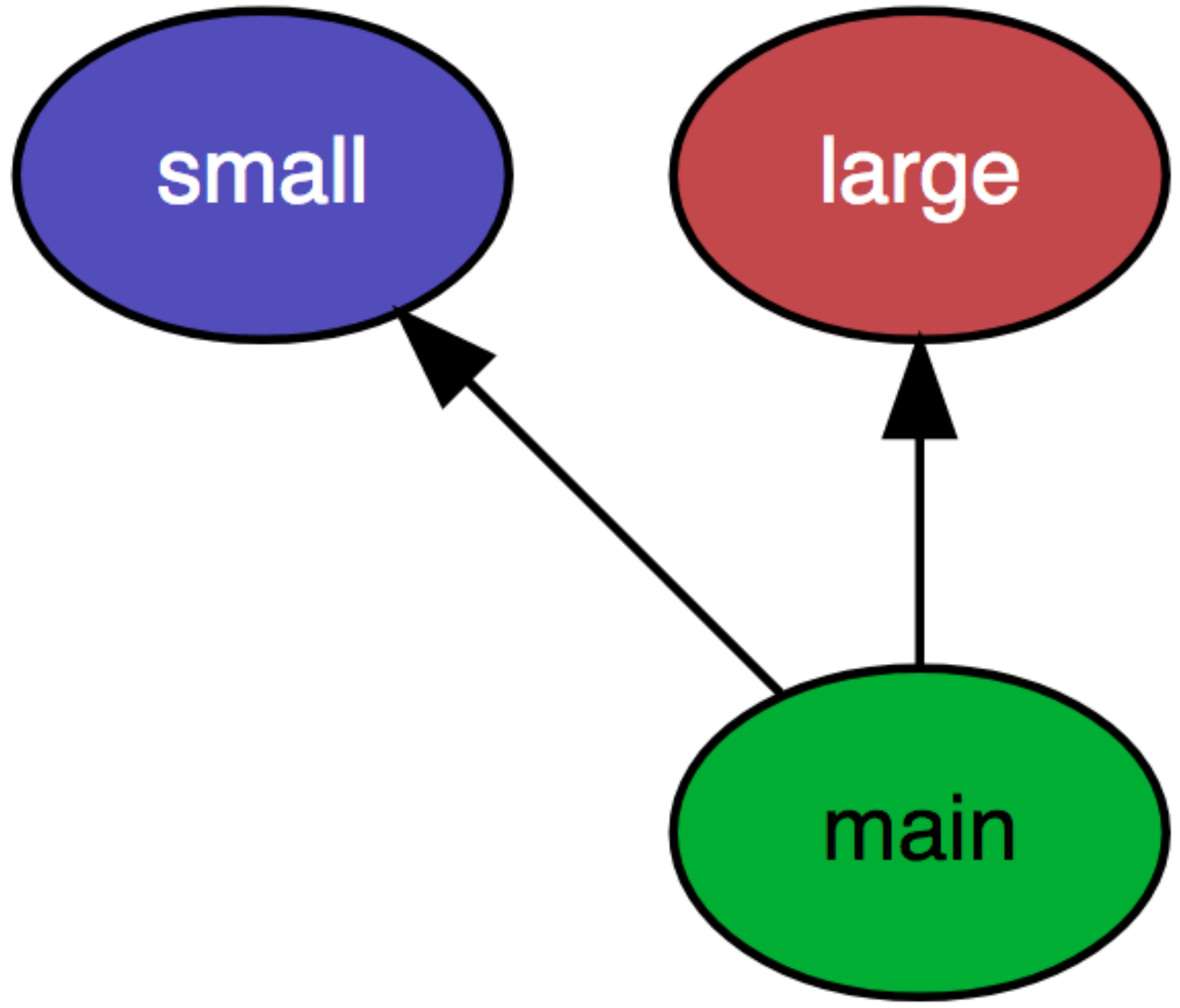
```python
import small
import large
print("Hi!")
```

small.py

```python
import time
time.sleep(1)
```

large.py

```python
import time
time.sleep(10)
```



Visualising the data

Showing imports as a graph

main.py

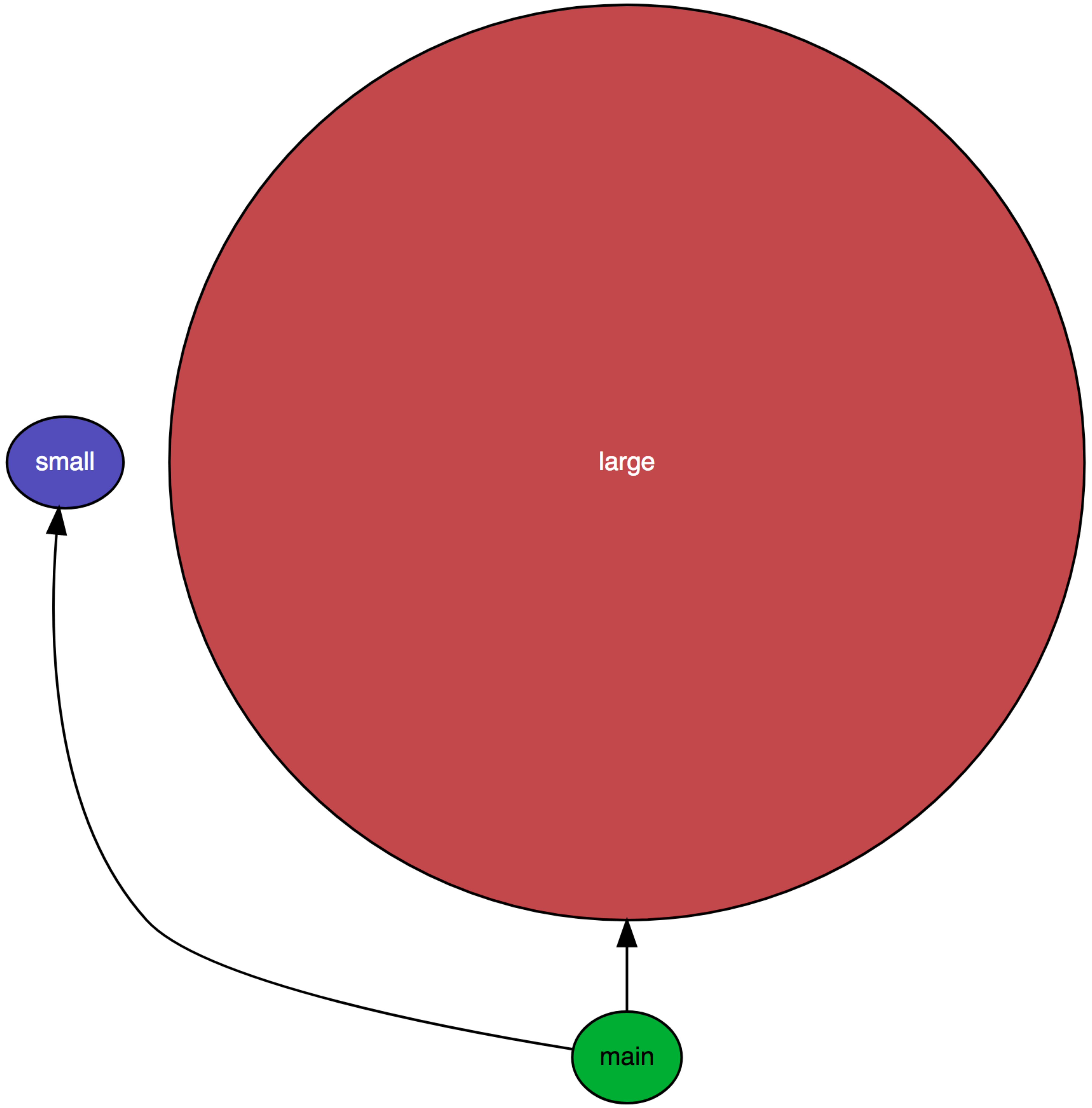
```python
import small
import large
print("Hi!")
```

small.py

```python
import time
time.sleep(1)
```

large.py

```python
import time
time.sleep(10)
```



Determining import cost

A more complicated example

Determining import cost

A more complicated example

main.py

```python
import one
import two
```

Determining import cost

A more complicated example

main.py

```python
import one
import two
```

one.py

```python
from funcs import *
from constants import PI
x = complicated(PI)
confusing(oh_no_help(x))
```

dport.me/pycon.pdf

Determining import cost

A more complicated example

main.py

```python
import one
import two
```

one.py

```python
from funcs import *
from constants import PI
x = complicated(PI)
confusing(oh_no_help(x))
```

two.py

```python
from constants import PI
print(f"pi: {PI}")
```

dport.me/pycon.pdf

Determining import cost

A more complicated example

main.py

```python
import one
import two
```

one.py

```python
from funcs import *
from constants import PI
x = complicated(PI)
confusing(oh_no_help(x))
```

two.py

```python
from constants import PI
print(f"pi: {PI}")
```

funcs.py

```python
time.sleep(0.3)

def complicated(a):
    pass
def confusing(b):
    pass
def oh_no_help(c):
    pass
```

Determining import cost

A more complicated example

main.py

```python
import one
import two
```

one.py

```python
from funcs import *
from constants import PI
x = complicated(PI)
confusing(oh_no_help(x))
```

two.py

```python
from constants import PI
print(f"pi: {PI}")
```

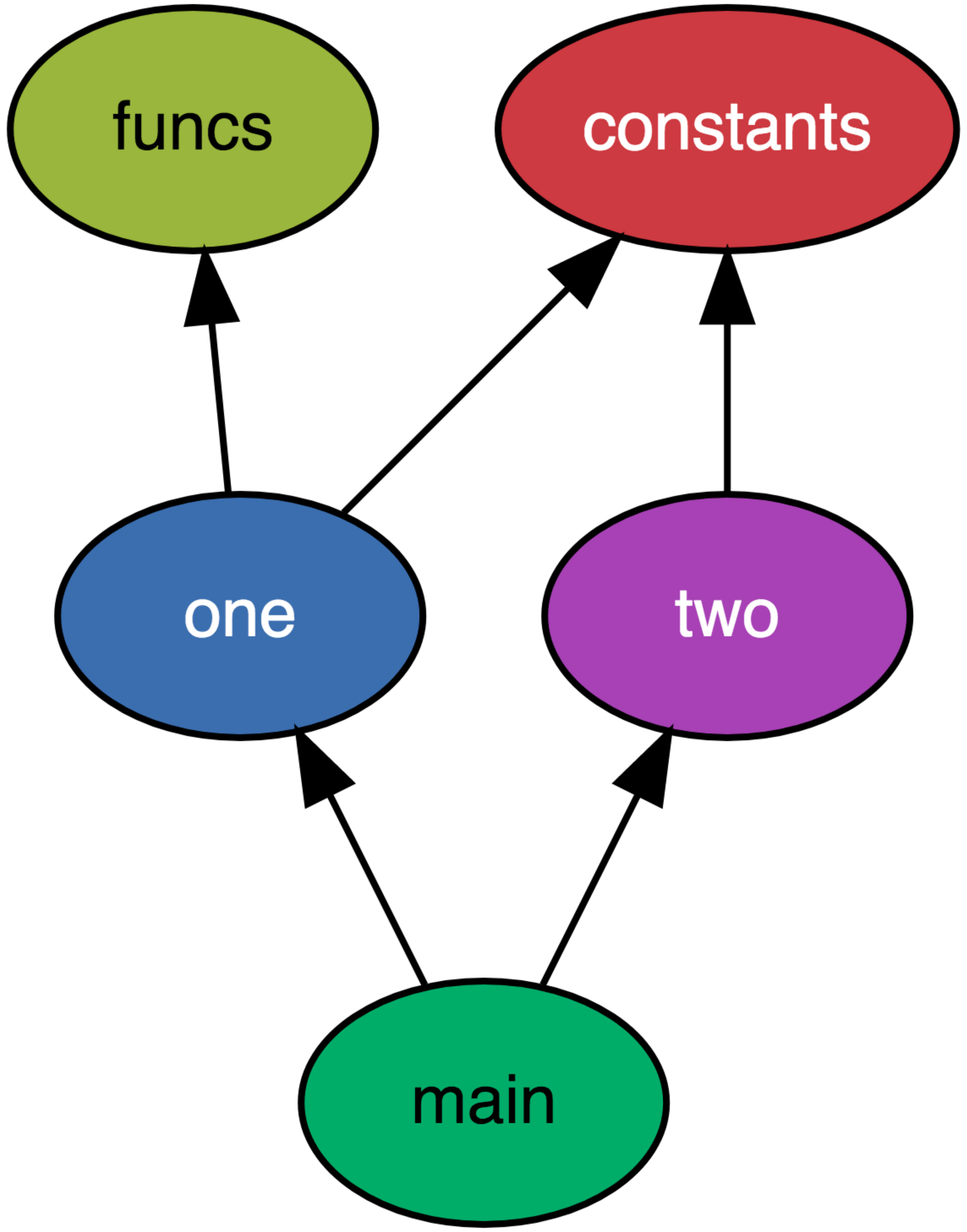
constants.py

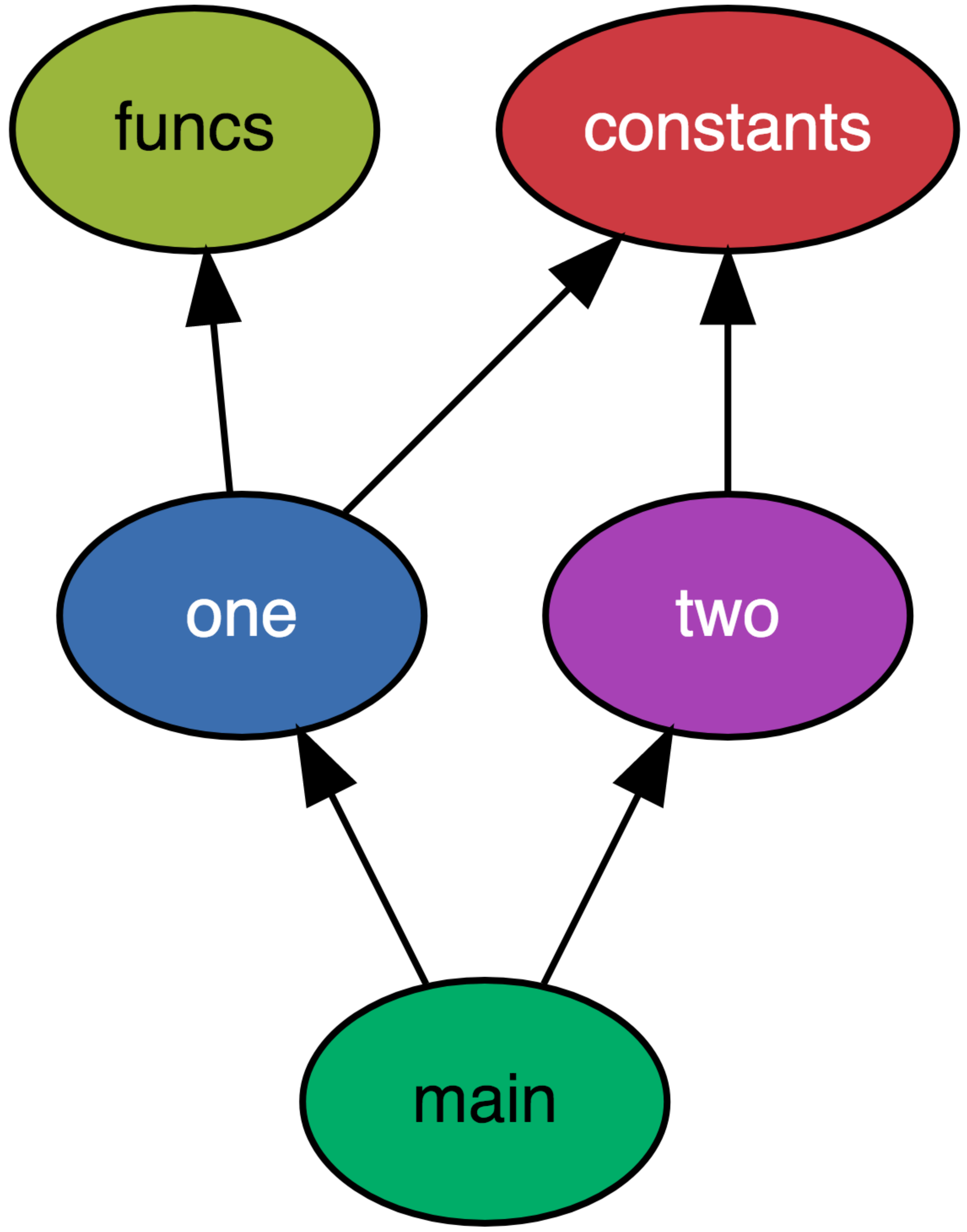
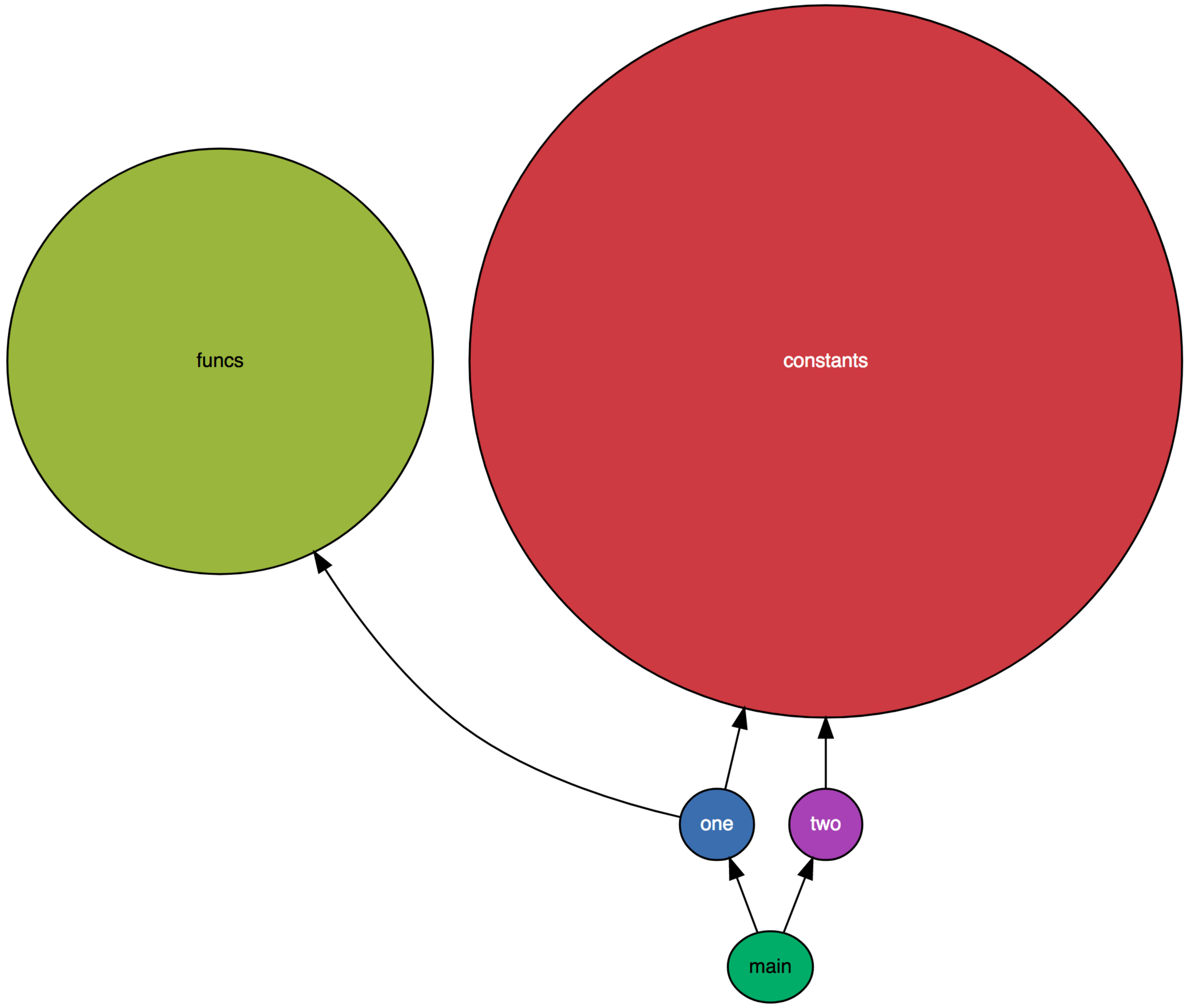
```python
time.sleep(0.5)
PI = 3.14
```

funcs.py

```python
time.sleep(0.3)

def complicated(a):
    pass
def confusing(b):
    pass
def oh_no_help(c):
    pass
```


funcs
constants
one
two
main


funcs
constants
one
two
main

funcs
constants
one
two
main

Determining import cost

Connectedness metrics

```
$ pydeps main.py --connectedness
  2 constants
  1 funcs
  1 one
  1 main
  1 two
  0 __main__
```

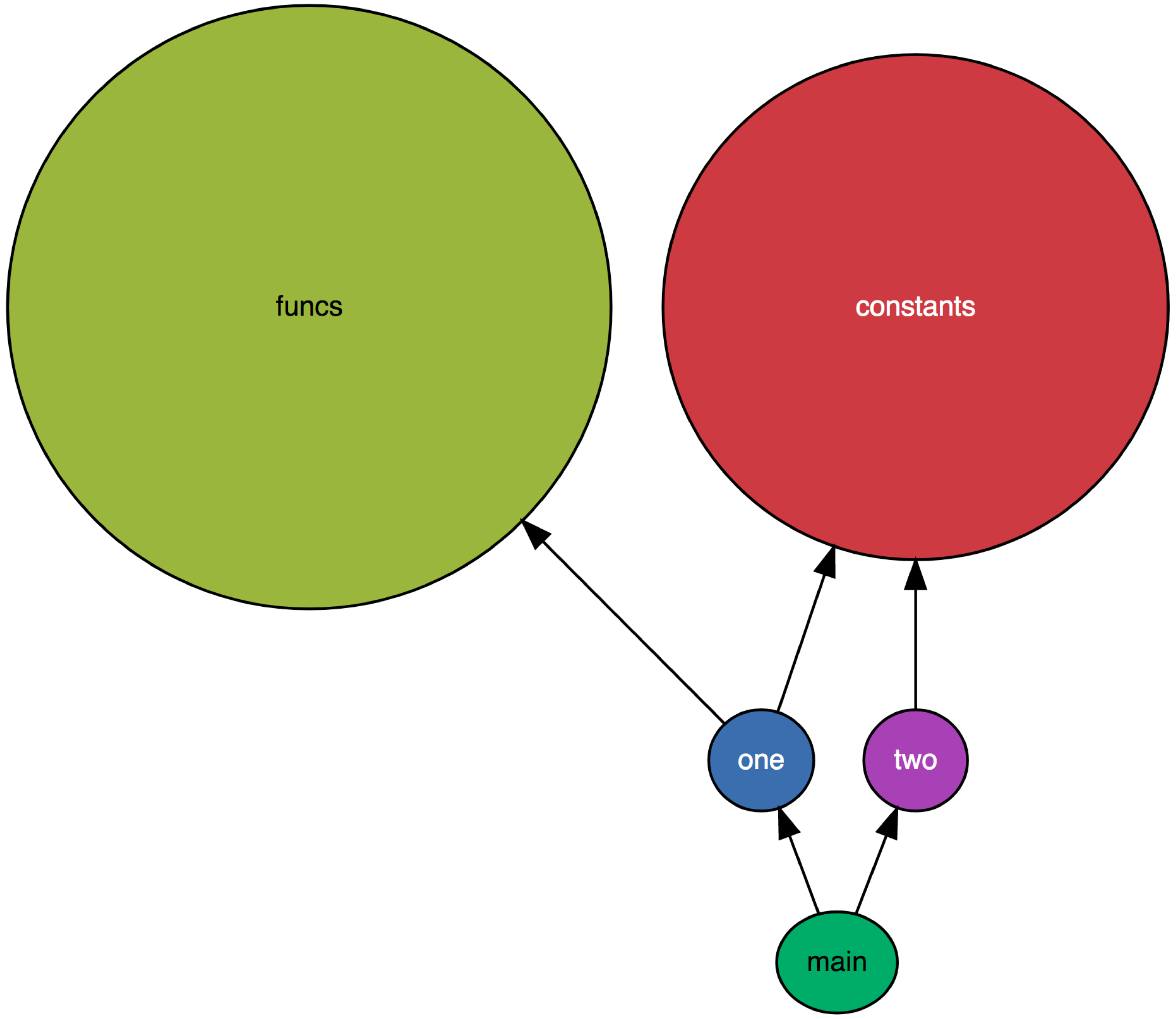
Determining import cost

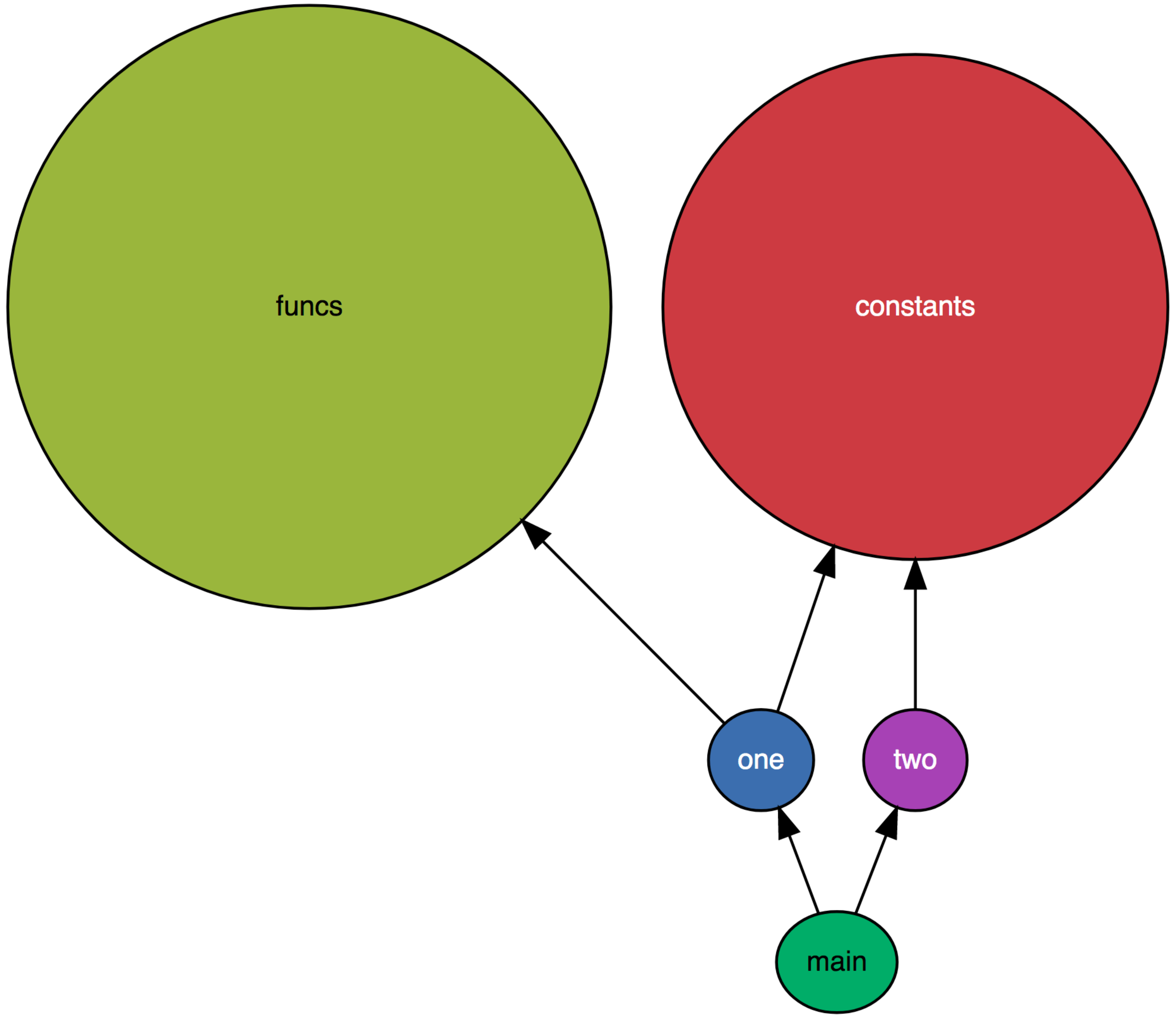
Proposing a node cost function with connectedness

Determining import cost

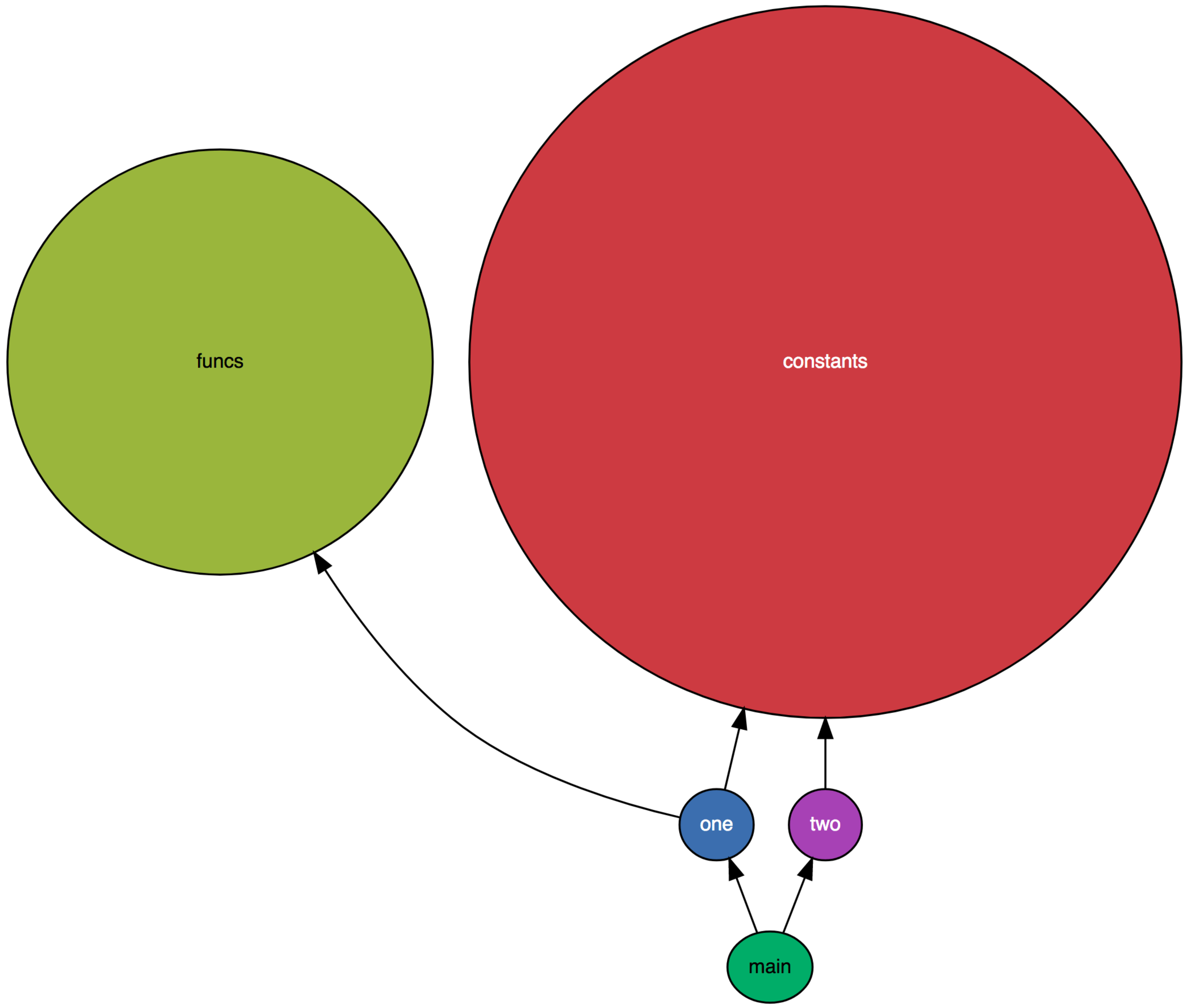
Proposing a node cost function with connectedness

$$\frac{\texttt{<import time>}}{\texttt{<num times imported>}}$$


funcs
constants
one
two
main



Import time + Connectedness



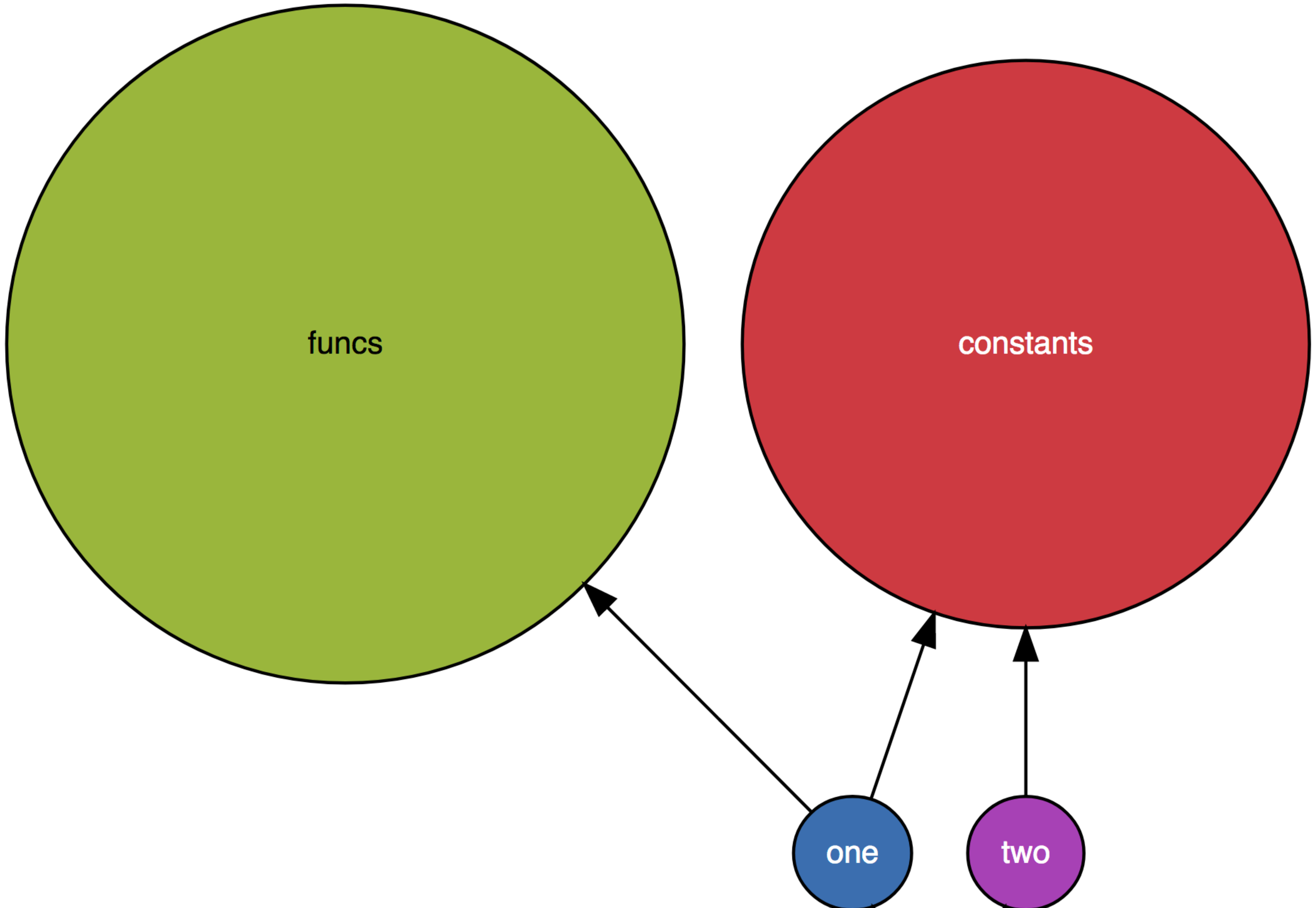
Import time

Visualising the data

Consider unique attribute access

Visualising the data

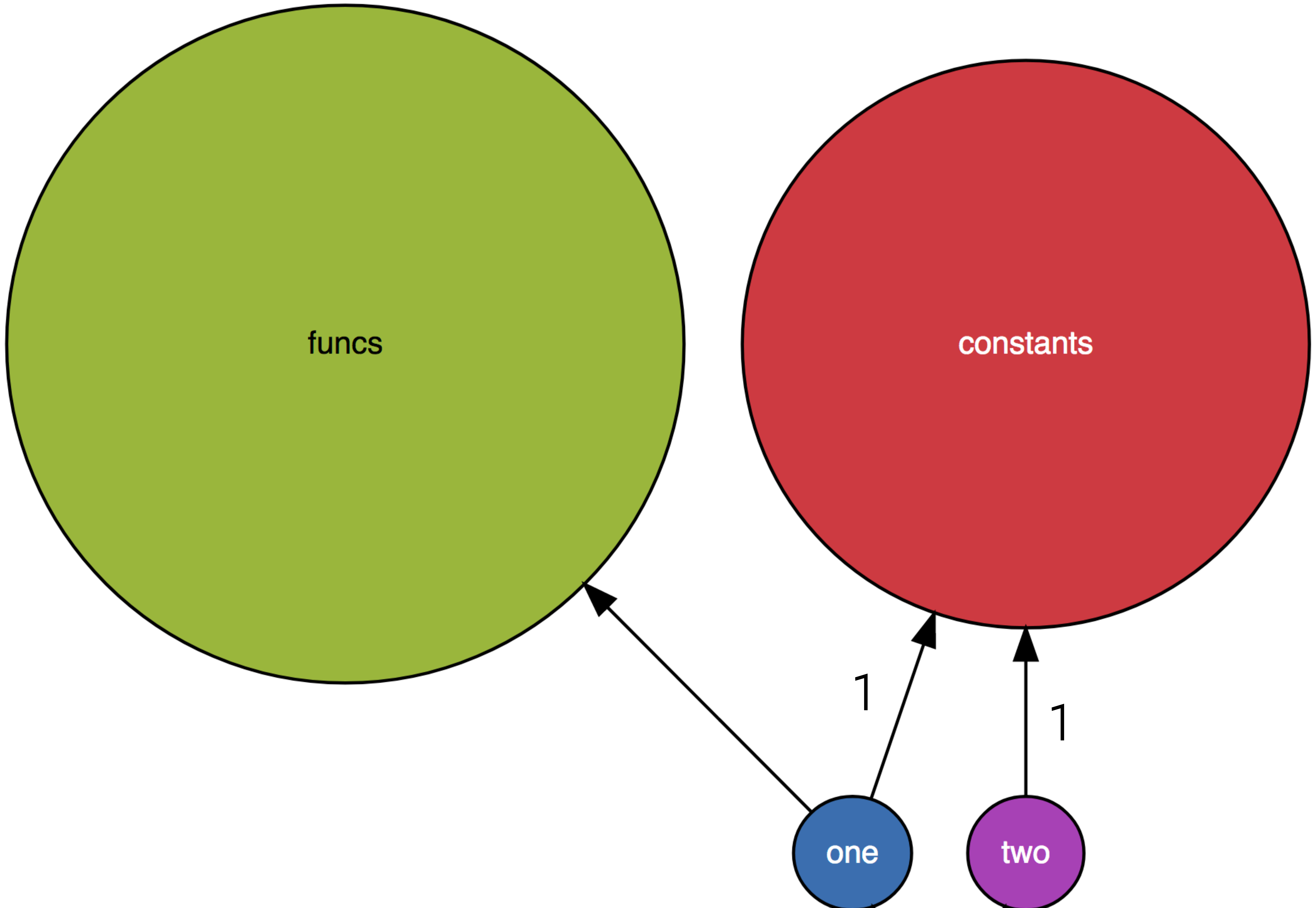
Consider unique attribute access



dport.me/pycon.pdf

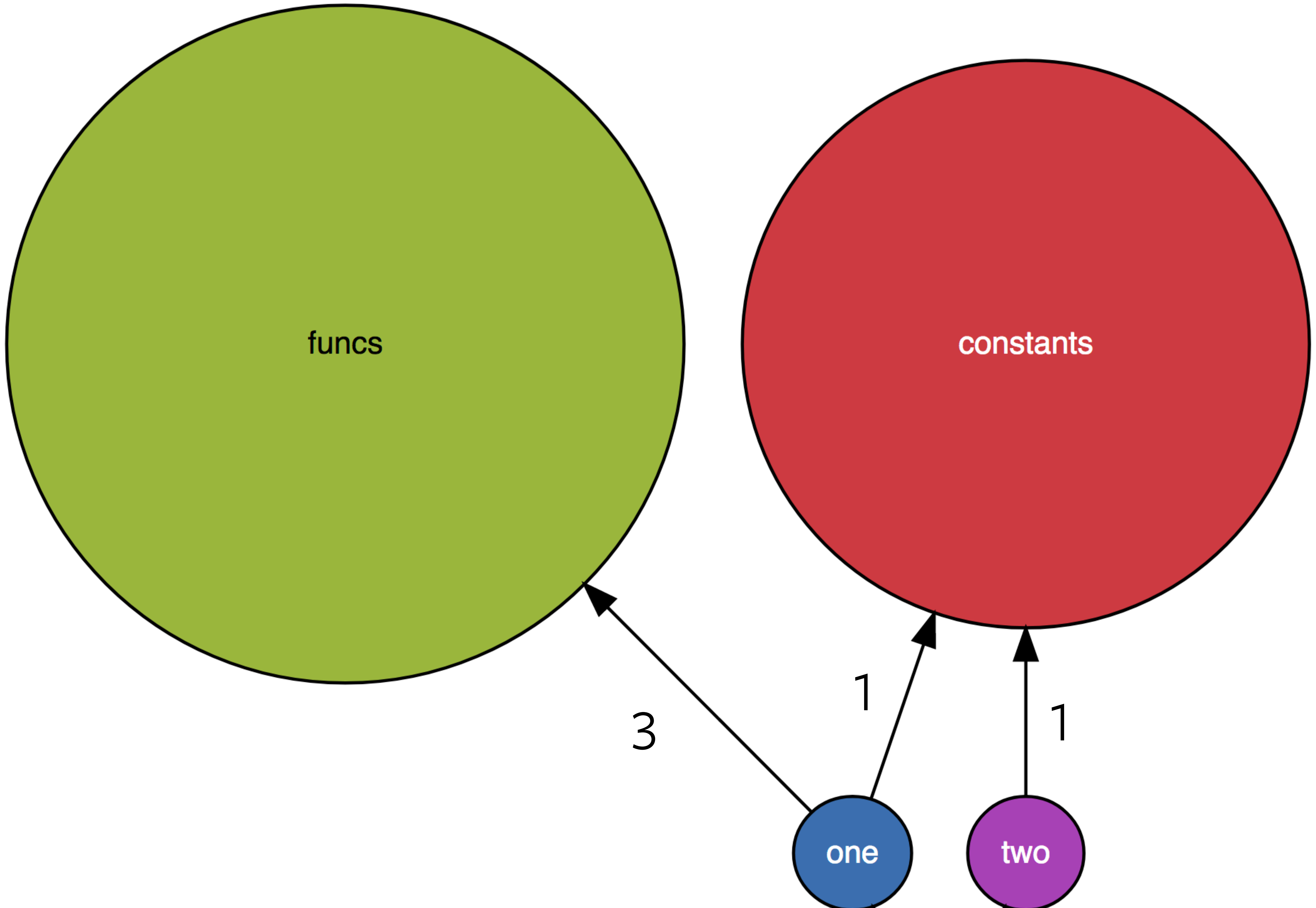
Visualising the data

Consider unique attribute access



Visualising the data

Consider unique attribute access



Visualising the data

A node cost function with connectedness and unique attribute access

Visualising the data

A node cost function with connectedness and unique attribute access

$$\frac{\texttt{<import time>}}{\texttt{<num times imported> + <unique attributes accessed>}}$$

Visualising the data

A node cost function with connectedness and unique attribute access

$$\frac{\texttt{<import time>}}{\texttt{<num times imported> + <unique attributes accessed>}}$$

constants.py

$$\frac{0.5}{2 + 1} = 0.166$$

Visualising the data

A node cost function with connectedness and unique attribute access

$$\frac{\texttt{<import time>}}{\texttt{<num times imported>} + \texttt{<unique attributes accessed>}}$$

constants.py

$$\frac{0.5}{2 + 1} = 0.166$$

funcs.py

$$\frac{0.3}{1 + 3} = 0.075$$

Visualising the data

A node cost function with connectedness and unique attribute access

$$\frac{\texttt{<import time>}}{\texttt{<num times imported> + <unique attributes accessed>}}$$

👀 constants.py 👀

$$\frac{0.5}{2 + 1} = 0.166$$

funcs.py

$$\frac{0.3}{1 + 3} = 0.075$$

Improving import time

Finally!

Improving import time

Finally!

- Restructure imports

Improving import time

Finally!

- Restructure imports
- Avoid doing work at module level

Improving import time

Finally!

- Restructure imports
- Avoid doing work at module level
  - Try lazy instantiation

dport.me/pycon.pdf

```python
class Expensive:
    def beegyoshi(self):
        return "!!!"
```

```python
class Expensive:
    def beegyoshi(self):
        return "!!!"

e = Expensive()
```

dport.me/pycon.pdf

```python
class Expensive:
    def beegyoshi(self):
        return "!!!"

class LazyWrapper:
    def __init__(self, clazz, *args, **kwargs):
        self.clazz = clazz
        self.args = args
        self.kwargs = kwargs
        self.instance = None

    def __getattr__(self, attr):
        if self.instance is None:
            self.instance = self.clazz(*self.args, **self.kwargs)
        return getattr(self.instance, attr)

e = LazyWrapper(Expensive)
```

Improving import time

Finally!

- Restructure imports
- Avoid doing work at module level
  - Try lazy instantiation

Improving import time

Finally!

- Restructure imports
- Avoid doing work at module level
  - Try lazy instantiation
- Lazy imports

Improving import time

Finally!

- Restructure imports
- Avoid doing work at module level
  - Try lazy instantiation
- Lazy imports
  - Import module within function that uses it

Improving import time

Finally!

- Restructure imports
- Avoid doing work at module level
  - Try lazy instantiation
- Lazy imports
  - Import module within function that uses it
  - Demand Import (https://pypi.org/project/demandimport/)

Improving import time

Demand import

main.py

dport.me/pycon.pdf

Improving import time

Demand import

main.py

```python
import one
import two
```

Improving import time

Demand import

main.py

```python
import demandimport
demandimport.enable()

import one
import two
```

dport.me/pycon.pdf

Improving import time

Demand import

main.py

```python
import demandimport
demandimport.enable()

import one
import two
```

```
$ python main.py
Delaying import of one for __main__ (level 0) situation #1
Delaying import of two for __main__ (level 0) situation #1
```

Improving import time

Demand import

main.py

```python
import demandimport
demandimport.enable()

import one
import two
```

dport.me/pycon.pdf

Improving import time

Demand import

main.py

```python
import demandimport
demandimport.enable()

import one
import two

print(one.x)
```

Improving import time

Demand import

```
$ python main.py
Delaying import of one for __main__ (level 0) situation #1
Delaying import of two for __main__ (level 0) situation #1
Triggered to import one for __main__
Delaying import of time for funcs (level 0) situation #1
Triggered to import time for funcs
Delaying import of time for constants (level 0) situation #1
Triggered to import time for constants
None
```

Summary

We made it!

Summary

We made it!

- How to get import times in 3.7 and < 3.7

Summary

We made it!

- How to get import times in 3.7 and < 3.7
- Different methods for visualising this data

Summary

We made it!

- How to get import times in 3.7 and < 3.7
- Different methods for visualising this data
  - As a tree and a graph

Summary

We made it!

- How to get import times in 3.7 and < 3.7
- Different methods for visualising this data
  - As a tree and a graph
  - Taking into account additional parameters

Summary

We made it!

- How to get import times in 3.7 and < 3.7
- Different methods for visualising this data
  - As a tree and a graph
  - Taking into account additional parameters
- Suggestions for how to fix it

Summary

We made it!

- How to get import times in 3.7 and < 3.7
- Different methods for visualising this data
  - As a tree and a graph
  - Taking into account additional parameters
- Suggestions for how to fix it
  - Import restructuring, lazy instantiation and importing

Thanks to these great folks 😍😍😍😍😍

- Lachlan for helping me prepare
- The Pycon AU team!!
- All of you lovely people 😊

Questions?

Twitter	Website	Github
@banool1	dport.me	github.com/banool

Tools you (yes you!) can use

Tool / Resource	Link	Use	Version
`-X importtime`	Native in 3.7	Get import time data	3.7+
import_times	github.com/banool/import_times	Get import time data (not as good)	3.4+
Tuna	github.com/nschloe/tuna	Show import time data in a flame graph / icicle chart format	3.7+
Pydeps	github.com/thebjorn/pydeps	Show import graph. Import time functionality only on my fork right now	2.7+
demandimport	pypi.org/project/demandimport/	Delay all module import to first attribute access	2.7+
All code used in this talk + more	github.com/banool/pycon-au-2019	See the many rabbitholes I fell down while writing this talk	3.7+

facebook

BLAH

BLAAAAHHHHHHH

- BLAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAHHH!!!!