eCAMI Core Refactoring

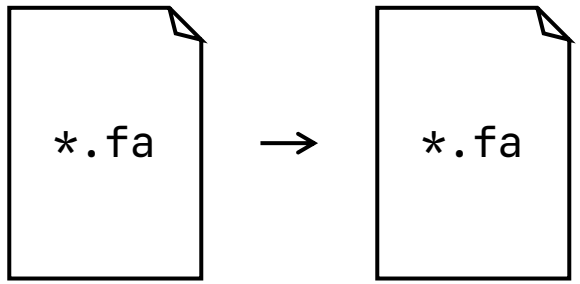
Shane Oh

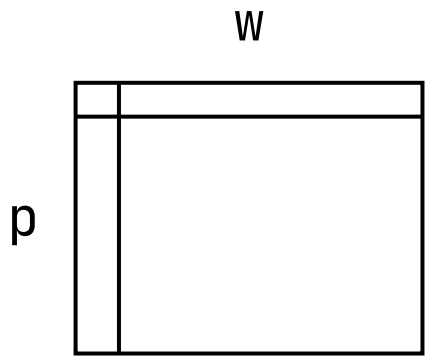
Email : ohshane71@gmail.com
Mobile: +82 10 8932 0939
Git   : github.com/ohshane71

Project Repository

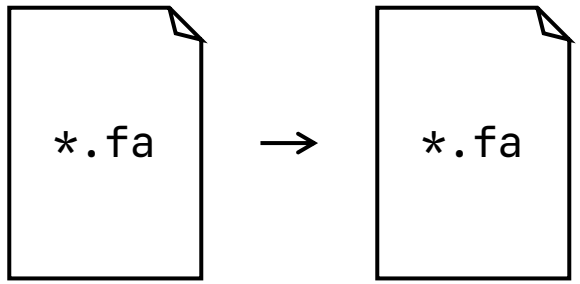
https://github.com/ohshane71/eCAMI
https://github.com/ohshane71/eCAMI/tree/dev

Pipeline

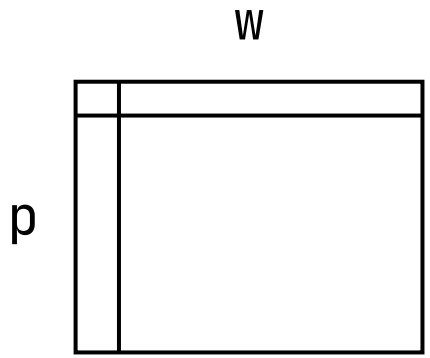




Pipeline



proteinIO



profast

proteinIO

File IO

Preprocessing

Shape Check

profast

Fit

Predict

Evaluate

proteinIO

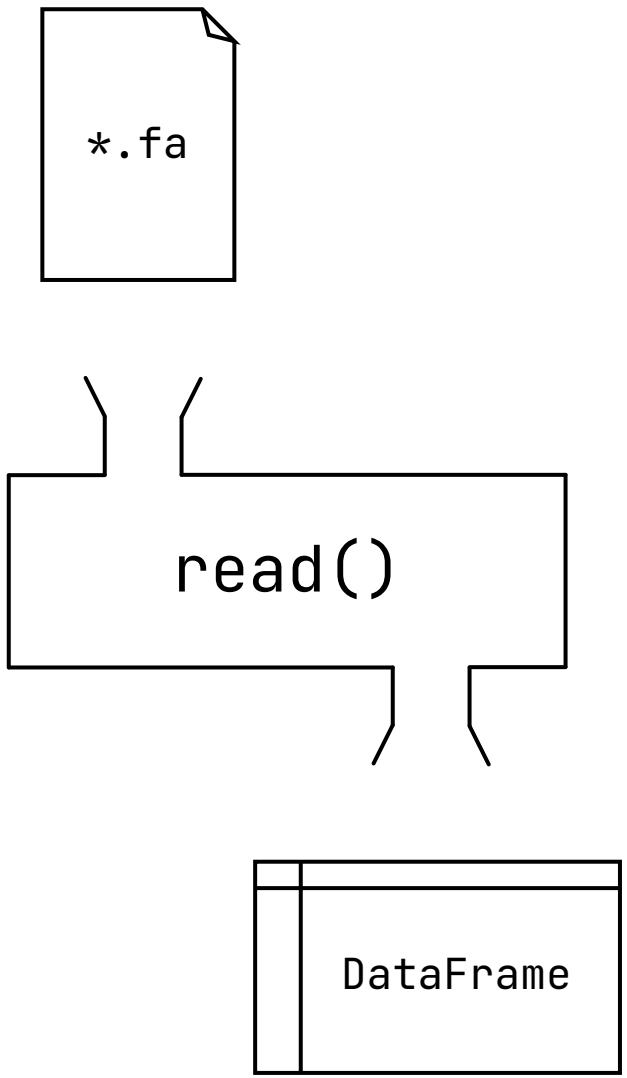
```
import proteinIO.fastaIO
import proteinIO.preprocessing
import proteinIO.protein
```

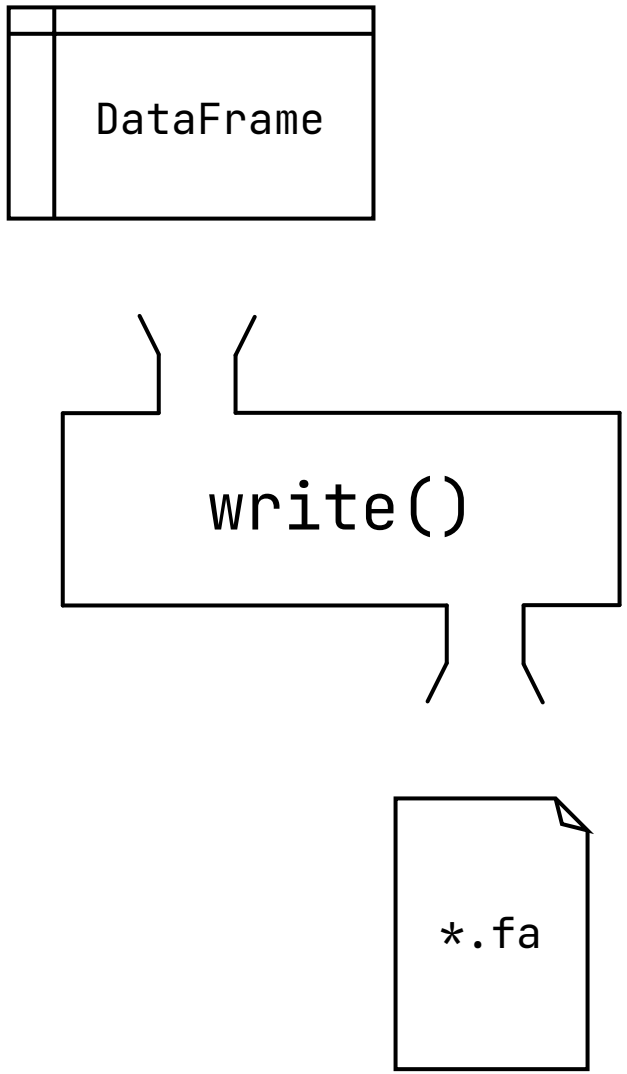
```
import proteinIO.fastaIO
import proteinIO.preprocessing
import proteinIO.protein
```

```
import proteinIO.fastaIO
import proteinIO.fastqIO
import proteinIO.    IO
```

```
from proteinIO.fastaIO import *


read()
write()
```

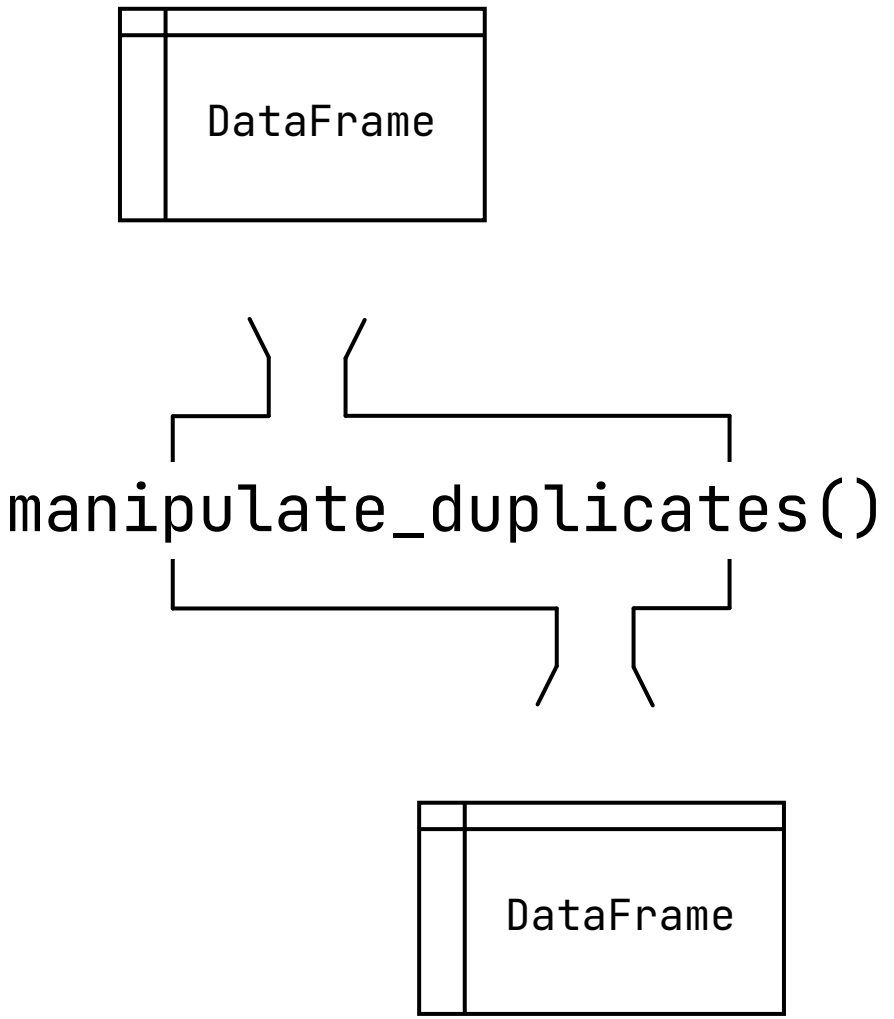

*.fa
read()
DataFrame


DataFrame
write()
*.fa

```
import proteinIO.fastaIO
import proteinIO.preprocessing
import proteinIO.protein
```

```
from proteinIO.preprocessing import *


manipulate_duplicates()
series2object()
shape()
```


DataFrame
manipulate_duplicates()
DataFrame

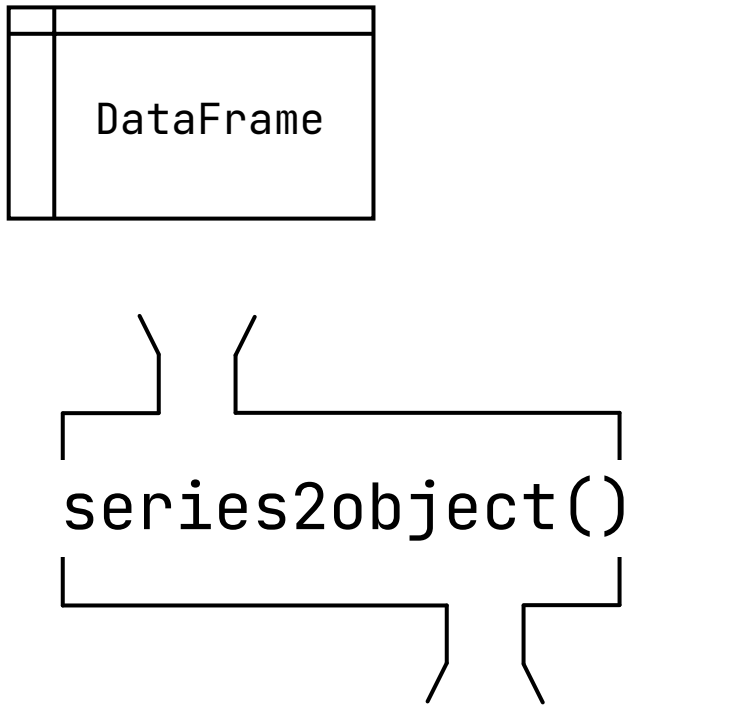
manipulate_duplicates()

@param

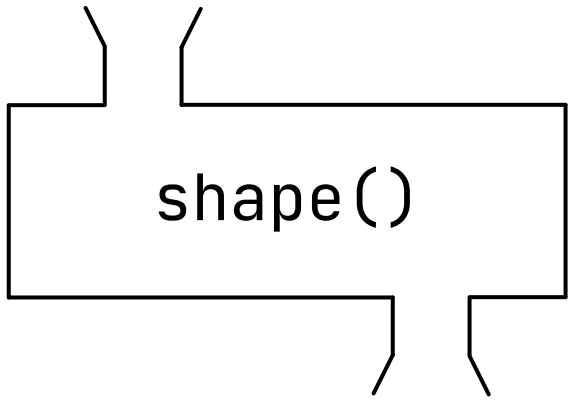
keep='union'

```
>A9.9|GT5|GT5_1
MLSVVVPVYNEEKNVEELVK
>A9.0|GT5|GT5_4|CBM48
MLSVVVPVYNEEKNVEELVK
```

```
>A9.9|GT5_1|GT5_4|CBM48
MLSVVVPVYNEEKNVEELVK
```


DataFrame
series2object()
[p, p, … , p]

[p, p, … , p]



{ family dictionary }

Nested dictionary

```
{
  “GH13” :
  {
    “_count”    : 3,
    “_ex_count” : 1,
    “_elements” : [2, 7, 109],
    “11”        :
     {
       “_count”    : 2,
       “_ex_count” : 2,
       “_elements” : [2, 109],
     },
  }
}
```

```
import proteinIO.fastaIO
import proteinIO.preprocessing
import proteinIO.protein
```

```
from proteinIO.protein import *

Protein()
```

```
p = Protein()

p.id        = “A9.1”
p.families = [[“GH5”],[“GH4”, “1”]]
p.sequence = “MLSVVVPVYNEEKNVEELVK”

# __call__ -> (__iter__, __next__)
for i, word in iter( p(k=8) ):
    pass
```

profast

```
import profast.kmer.KMerClassifier
```

```
from profast.kmer.KMerClassifier import *

KMerClassifier()
```

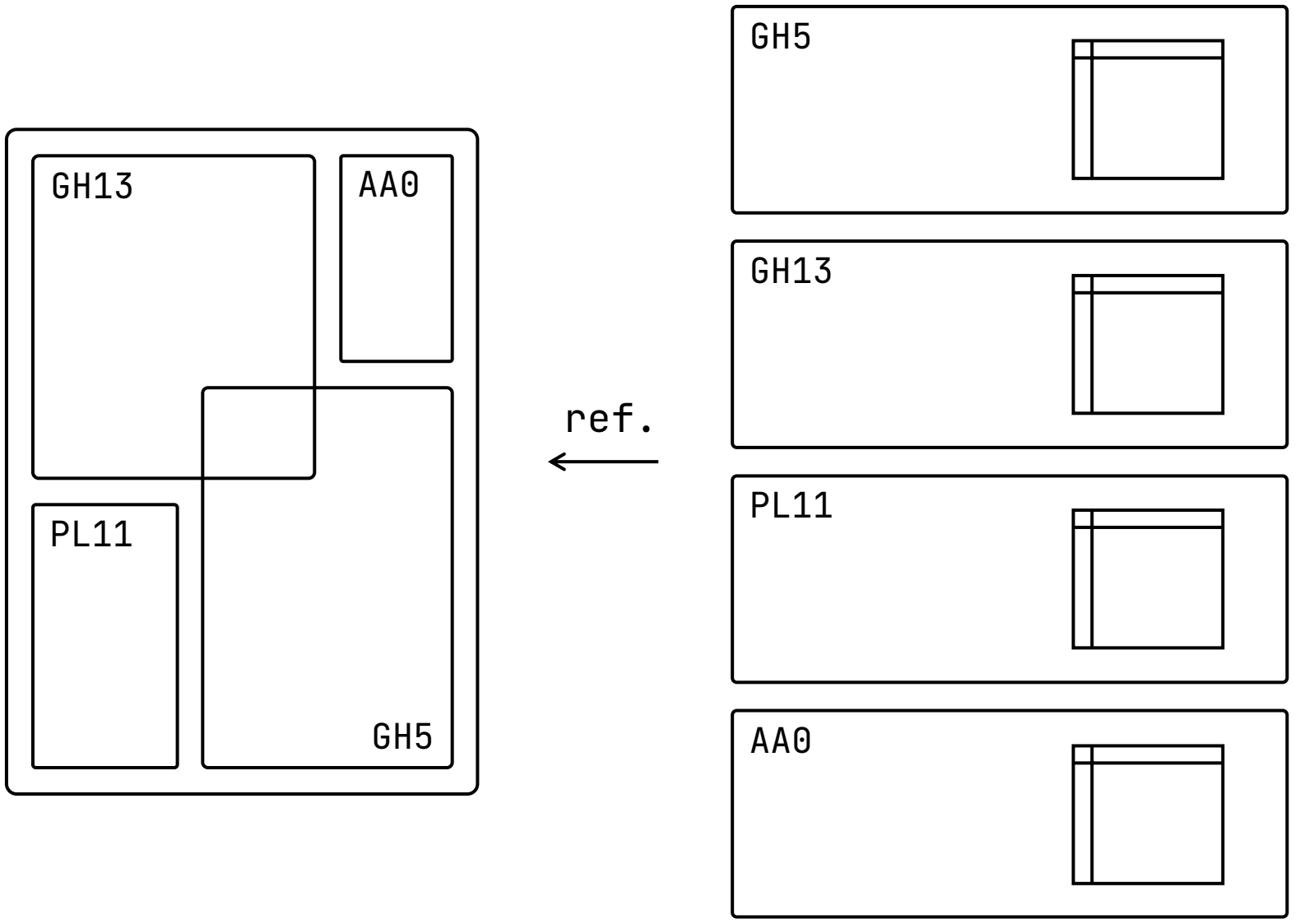
```
kmc = KMerClassifier()

kmc.alpha = 2
kmc.beta  = 0.9


kmc.fit(X, shape, families, k)

kmc.X        = [p, p, … , p]
kmc.shape    = { family dictionary }
kmc.families = [“GH5”, “GH13”, “PL11”, “AA0”]|[“*”]
kmc.k        = 8
```

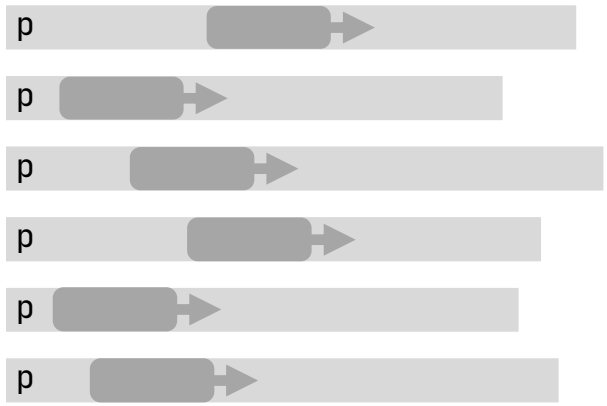
Multiprocessing



Threading

GH5

GH13



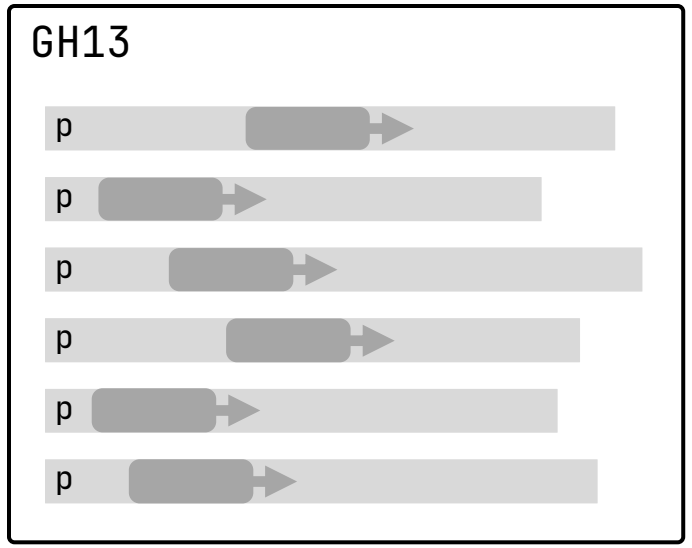
PL11

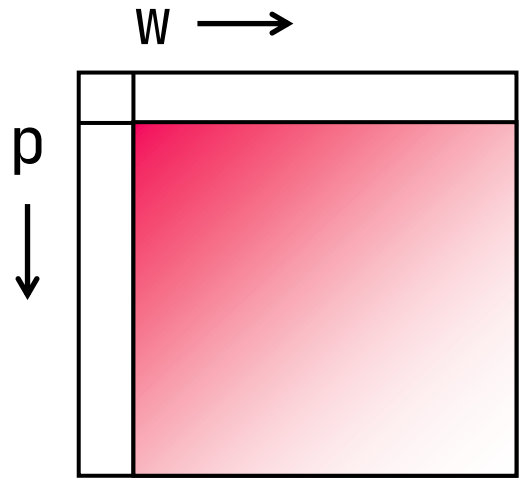
AA0

shared matrix per family

GH5

GH13





PL11

AA0