Optimisation of the Ganga toolkit in terms of memory consumption and persistent storage.

**Mentors:** Prof. Ulrik Egede, Dr. Alexander Richards, Dr. Mark Smith

**Student Developer:** Ishan Rai (IIT, Roorkee)
Overview

Resource Profiling:

Implemented decorators to:

- profile CPU Consumption: Used python package cProfiler
- profile Memory consumption: Used python package memory_profile
- count number of function calls.
<method 'run' of '_Profiler' objects> at 0x99f780
<__instancecheck__>
<__new__>
<__init__>
<__contains__>
<__deepcopy__>
<__new__ of type object at 0x99f780>
<__getattribute__>
<__instancecheck__>
<__new__>
<__contains__>
<__deepcopy__>
<__new__ of type object at 0x99f780>
<__getattribute__>
**Memory Optimization:**

**Flyweight Pattern:** Flyweight pattern is one of the structural design patterns. This pattern provides ways to decrease object count thus improving application required objects structure.

```java
FlyweightFactory
GetFlyweight(key) →
|
| flyweights
|
|
Flyweight
Operation(extrinsicState)
|
|
ConcreteFlyweight
Operation(extrinsicState)
IntrinsicState
|
|
UnsharedConcreteFlyweight
Operation(extrinsicState)
allState
|
|
Client
```

**Copy-On-Write:** Copy-on-write is a technique to efficiently copy data resources in a computer system. If a unit of data is copied but not modified, the "copy" can exist as a reference to the original data. Only when the copied data is modified is a copy created, and new bytes are actually written.
Implementation Details

1. We use a HashMap that stores reference to the object which have already been created.
2. Every object is associated with a key.
3. When a client wants to create an object, they simply have to pass a key associated with it.
4. If the object has already been created we simply get the reference to that object else it creates a new object and then returns it reference to the client.
5. Both parent and child processes to initially share the same object in memory.
6. If either process modifies a shared object, only then is the object copied.
Future Work:

- Extend the principle of Copy On Write to persistent storage
- Continue contributing to Ganga

Acknowledgement

I. Prof. Ulrik Egede, Dr. Alexander Richards and Dr. Mark Smith
II. Design Patterns: Elements of Reusable Object-Oriented Software (Erich Gamma, John Vlissides, Ralph Johnson, and Richard Helm)