1. Compilation



1. compile application (server) code

Text

Description automatically generated

1. compile console-client code

Text

Description automatically generated

1. compile batch-client code

Text

Description automatically generated

1. run server

Text

Description automatically generated

Note:

1. Remember change the program path and data path in run.sh to yours.
2. The server-side program may exit abnormally, due to invalid query string or due to bugs of the query processing algorithms. In this case, you may either use the ipcrm command or run "./kill.py".
3. run console-client version

Notes:

You can run client only when server side is running.

To terminate the server program from client, type "server\_exit".

To terminate a client program from console, type "exit".

The results will be in “experiments/ol\_out” folder.

Text

Description automatically generated

1. run batch-client version

Text

Description automatically generated

**### API ###**

Our Tthinker API is similar like our Gthinker API:

<https://yanlab19870714.github.io/yanda/gthinker/api.html>

In your application code, you should implement the following function.

class Comper:

1. virtual int toQuery(string& line)
2. virtual bool task\_spawn(QueryT q)
3. virtual bool is\_bigTask(TaskT \*task)
4. virtual void compute(ContextT &context)
5. virtual bool is\_bigTask(TaskT \*task)

You may also should implement the your load\_data() in your Worker class.

There is no Trimmer class and Aggregator class currently in our Tthinker.