OpenQTSim Documentation

Release v0.5.0

Mark van Koningsveld, Joris den Uijl

Nov 13, 2020

Contents:

OpenQTSim is a python package for queueing theory simulations.

Welcome to OpenQTSim documentation! Please check the contents below for information on installation, getting started and actual example code. If you want to dive straight into the code you can check out our GitHub page or the working examples presented in Jupyter Notebooks.

CHAPTER 1

Installation

1.1 Stable release

To install OpenQTSim, run this command in your terminal:

```
# Use pip to install OpenQTSim
pip install openqtsim
```

This is the preferred method to install OpenQTSim, as it will always install the most recent stable release.

If you do not pip installed, this Python installation guide can guide you through the process.

1.2 From sources

The sources for OpenQTSim can be downloaded from the Github repo.

You can either clone the public repository:

```
# Use git to clone OpenQTSim
git clone git://github.com/TUDelft-CITG/OpenQTSim
```

Or download the tarball:

```
# Use curl to obtain the tarball
curl  -OL https://github.com/TUDelft-CITG/OpenQTSim/tarball/master
```

Once you have a copy of the source, you can install it with:

```
# Use python to install
python setup.py install
```

CHAPTER 2

OpenQTSim

This page lists all functions and classes available in the OpenQTSim modules. For examples on how to use these submodules please check out the Examples page, information on installing OpenQTSim can be found on the Installation page.

2.1 Submodules

The main components are the Model module and the Core module. All of their components are listed below.

2.2 openqtsim.arrival_process module

class openqtsim.arrival_process.**ArrivalProcess**(*symbol='M'*, *arr_rate=8*)
Bases: object

Arrival process class for use in the OpenQTSim package

get_IAT(*customer_nr=[]*)
Return the inter arrival time based on the inter arrival time distribution or deterministic list

2.3 openqtsim.customer module

class openqtsim.customer.**Customer**(*Env*, *Sim*)
Bases: object

Customer class for use in the OpenQTSim package

move(*IAT*, *AT*)
" Method to move Customer through the system

2.4 openqtsim.queue module

class openqtsim.queue.**Queue**(*A=<openqtsim.arrival_process.ArrivalProcess object>*, *S=<openqtsim.service_process.ServiceProcess object>*, *c=1*, *K=inf*, *N=inf*, *D='FIFO'*)

Bases: object

Queueing class based on Kendall's notation, in which: - A is the arrival process - S is the service time distribution - c is the number of servers - K is the number of places in the system - N is the calling population - D is the queue discipline

kendall_notation
Return queue name according to the Kendall notation.

occupancy_to_waitingfactor(*utilisation=0.3*, *nr_of_servers_to_chk=4*, *poly_order=6*)
Waiting time factor (E2/E2/n or M/E2/n) queueing theory using 6th order polynomial regression)

populate(*Env*, *Sim*)
While the simulation time does not exceed the maximum duration, generate customers according to the distribution of the arrival process to populate the queue

waitingfactor_to_occupancy(*factor=0.3*, *nr_of_servers_to_chk=4*, *poly_order=6*)
Waiting time factor (E2/E2/n or M/E2/n) queueing theory using 6th order polynomial regression)

2.5 openqtsim.service_process module

class openqtsim.service_process.**ServiceProcess**(*symbol='M'*, *srv_rate=9*)
Bases: object

Server process class for use in the OpenQTSim package

get_ST(*server*, *customer_nr=[]*)
Return the inter arrival time based on the inter arrival time distribution or deterministic list

2.6 openqtsim.simulation module

class openqtsim.simulation.**Simulation**(*queue*, *max_arr=100*, *priority=False*, *seed=None*)
Bases: object

A discrete event simulation that simulates the queue. - queue is a queue based on the queue class - seed is a random seed to have retraceable simulations

get_stats()
Post processing of logs to print basic simulation statistics

log_customer_state(*customer_id*, *IAT*, *AT*, *ST*, *TSB*, *TSE*, *ITS*, *s_id*)
the following items are logged per customer that enters the system: # c = customer id # IAT = inter arrival time # ST = service time # AT = arrival time # TSB = time service begins # TSE = time service ends # TCSS = time customer spends in the system # TCWQ = time customer waits in the queue # ITS = idle time of the server # s_id = id of server assigned to customer

log_system_state(*t*, *c_s*, *c_q*)
the following items are logged for the state of the system: # t = time (from start of simulation) # c_s = number of customers in the system # c_q = number of customers in the queue

plot_IAT_ST(*fontsize=20*)
Plot histograms of IAT's and ST's

plot_system_state(*fontsize=20*)
Plot number of customers in the system and in the queue as a function of time

return_log()
Return the log in the form of a pandas data frame.

run(*max_arr=1000*)
Run simulation

2.7 Module contents

Top-level package for queueing.

CHAPTER 3

Contributing

Contributions are welcome, and they are greatly appreciated! Every little bit helps, and credit will always be given.

You can contribute in many ways:

3.1 Types of Contributions

3.1.1 Report Bugs

Report bugs at https://github.com/TUDelft-CITG/OpenQTSim/issues.

If you are reporting a bug, please include:

- Your operating system name and version.
- Any details about your local setup that might be helpful in troubleshooting.
- Detailed steps to reproduce the bug.

3.1.2 Fix Bugs

Look through the GitHub issues for bugs. Anything tagged with "bug" and "help wanted" is open to whoever wants to implement it.

3.1.3 Implement Features

Look through the GitHub issues for features. Anything tagged with "enhancement" and "help wanted" is open to whoever wants to implement it.

3.1.4 Write Documentation

OpenQTSim could always use more documentation, whether as part of the official OpenQTSim docs, in docstrings, or even on the web in blog posts, articles, and such.

3.1.5 Submit Feedback

The best way to send feedback is to file an issue at https://github.com/TUDelft-CITG/OpenQTSim/issues.

If you are proposing a feature:

- Explain in detail how it would work.
- Keep the scope as narrow as possible, to make it easier to implement.
- Remember that this is a volunteer-driven project, and that contributions are welcome :)

3.2 Get Started!

Ready to contribute? Here's how to set up *OpenQTSim* for local development.

1. Fork the *OpenQTSim* repository on GitHub.
2. Clone your fork locally:

```
$ git clone git@github.com:your_name_here/OpenQTSim.git
```

3. Install your local copy into a virtualenv. Assuming you have virtualenvwrapper installed, this is how you set up your fork for local development:

```
$ mkvirtualenv openqtsim
$ cd openqtsim/
$ python setup.py develop
```

4. Create a branch for local development:

```
$ git checkout -b name-of-your-bugfix-or-feature
```

   Now you can make your changes locally.

5. When you're done making changes, check that your changes pass flake8 and the tests, including testing other Python versions with tox:

```
$ flake8 openqtsim tests
$ python setup.py test or py.test
$ tox
```

   To get flake8 and tox, just pip install them into your virtualenv.

6. The style of OpenQTSim is according to Black. Format your code using Black with the following lines of code:

```
$ black openqtsim
$ black tests
```

   You can install black using pip.

7. Commit your changes and push your branch to GitHub:

```
$ git add .
$ git commit -m "Your detailed description of your changes."
$ git push origin name-of-your-bugfix-or-feature
```

8. Submit a pull request through the GitHub website.

3.3 Pull Request Guidelines

Before you submit a pull request, check that it meets these guidelines:

1. The pull request should include tests.
2. If the pull request adds functionality, the docs should be updated. Put your new functionality into a function with a docstring, and add the feature to the list in README.rst.
3. The pull request should work for Python 3.4, 3.5 and 3.6, and for PyPy. Check CircleCI and make sure that the tests pass for all supported Python versions.

3.4 Tips

To run a subset of tests:

```
$ py.test tests.test_openqtsim
```

To make the documentation pages:

```
$ make docs # for linux/osx
```

For windows:

```
$ del docs\openqtsim.rst
$ del docs\modules.rst
$ sphinx-apidoc -o docs/ openqtsim
$ cd docs
$ make html
$ start explorer _build\html\index.html
```

3.5 Deploying

A reminder for the maintainers on how to deploy. Make sure all your changes are committed (including an entry in HISTORY.rst). Then run:

```
$ bumpversion patch # possible: major / minor / patch
$ git push
$ git push --tags
```

Travis will then deploy to PyPI if tests pass.

CHAPTER 4

Credits

4.1 Development Lead

- Mark van Koningsveld
- Joris den Uijl
- Fedor Baart

4.2 Contributors

So far the main contributions come from Joris den Uijl, Fedor Baart and Mark van Koningsveld.

CHAPTER 5

History

5.1 v0.5.0 (2020-03-11)

- Package made public (published via zenodo.org)
- Package still under development

5.2 v0.4.0 (2020-03-11)

- Package reorganised (code cleanup, multi thread engine added)

5.3 v0.3.0 (2020-02-06)

- Package reorganised (better output, examples included)

5.4 v0.2.0 (2020-02-02)

- Package renamed to OpenQTSim

5.5 v0.1.0 (2019-06-09)

- First version developed by Joris den Uijl

CHAPTER 6

Indices and tables

- genindex
- modindex
- search

Python Module Index

o

Index