CIW concepts and terminology

Network

Nodes

Nodes are the unit of network that do the services. Typically there are three nodes for each service:-

1. The Arrival Node:
2. Service Node
3. Exit Node

Arrival node

This is where customers are created. They are spawned here, and can baulk, be rejected or sent to a service node. Accessed using Q.nodes[0]

Service Node

This is where customers queue up and receive service. Accessed using: Q.nodes[1]

Exit Node

When customers leave the system, they are collected here. Then, when we wish to find out what happened during the simulation run, we can find the customers here. Accessed using: Q.nodes[-1]

We can even get the users at each node using Q.nodes[x].all_individuals, where x is the index of node.

And data of that particular customer like :-

```
ind = Q.nodes[-1].all_individuals[0]
>>> ind
Individual 2
>>> len(ind.data_records)
1

>>> ind.data_records[0].arrival_date
7.936299...
>>> ind.data_records[0].waiting_time
0.0
>>> ind.data_records[0].service_start_date
7.936299...
>>> ind.data_records[0].service_time
2.944637...
>>> ind.data_records[0].service_end_date
10.88093...
>>> ind.data_records[0].exit_date
10.88093...
```

Results

Once simulation is over we can get all results at once and use the data for analysis and show different charts using matplot or any other graphical interface as well.

recs = Q.get_all_records()

this method is used to gather all the data for all the customer at the end of simulation it consists of :-

- id_number - The unique identification number for a particular customer.
- customer_class - The number of that customer's customer class. If dynamic customer classes are used, this is the customer's previous class, before a new customer class is sampled after service.
- Node - The number of the node at which the service took place
- arrival_date - The date of arrival to that node, the date which the customer joined the queue.
- waiting_time - The amount of time the customer spent waiting for service at that node.
- service_start_date - The date at which service began at that node.
- service_time - The amount of time spent in service at that node.
- service_end_date - The date which the customer finished their service at that node.
- time_blocked - The amount of time spent blocked at that node. That is the time between finishing service at exiting the node.
- exit_date - The date which the customer exited the node. This may be immediately after service if no blocking occured, or after some period of being blocked.
- Destination - The number of the customer's destination, that is the next node the customer will join after leaving the current node. If the customer leaves the system, this will be -1
- queue_size_at_arrival - The size of the queue at the customer's arrival date. Does not include the individual themselves.
- queue_size_at_departure - The size of the queue at the customer's exit date. Does not include the individual themselves.

all the data are in the form of list and we can use list operations on them for better understanding as like to find the average , mean waiting time. Some simulation are hours long while some can take weeks or even months to complete like passport services. That's why date field is also included in the results.

server_utilisation

we can look at the server utilization of a Node. This is the average utilization of each server, which is the amount of time a server was busy (with a customer), divided by the total amount of time the server was on duty.

Correct way to utilize ciw for maximizing the accuracy

There are some facts that we need to consider when we run any simulation model.

- In real life any event does not start from the beginning as like for banking system there will be some pending work prior to the day, some work come on that day and at the end of the day some works are even not completed.
- In simulation model everything starts as zero for simulation day and at the end of the simulation unfinished work will be discarded.

So in order to maximize the similarity between the simulation and real word we can do the following things.

- Warm up period – we run simulation for a couple of minutes before taking the notes of simulation so that the simulation will not start from the idle state and can have some backlogs to replicate real life events.
- Cool off period - even at the end of simulation we run simulation for couple minutes so that the existing pending services can be taken care of.
- Trails – we run simulation over and over to include the randomness and to get a generalized outcome. Trails can be run in two ways :-
    - Same parameters but different seeds – to find the generalized answers of some particular simulation
    - Same seeds but different parameter - how simulation will change when we increase or decreases the load or servers.

Network of queues

In real life there are more than one service connected with each other for an example in banking some people go to cash withdrawal, some to deposit, but there can be also a case where one withdraw the cash and deposited to some other account. So basically here two services that works individually and also connected sometimes.

So in this case we pass list in the network and create a routing based on the inter-arrival time service time. A routing matrix is an matrix (where is the number of nodes in the network) such that the element corresponds to the probability of transitioning to node after service at node . In Python, we write this matrix as a list of lists.

[[0 0.3 0.7],

[0 0 1]

[0 0 0]]

Restricted network

Restricted networks are those network which is having a fixed queue size that is a correct representation of the service as we can have a limited size queue in real life. In restricted network we pass one extra parameter that is queue size.

ciw.dists.Deterministic(value=4.0)

if the interval is fixed like event occurs at every 4 second we can use deterministic distribution there.

ciw.dists.Uniform(lower=3, upper=5)

it will give a time between lower and upper limit.

Different Classes of customer

In real life there can be different classes of customers, like in hospital the classes can be child, general, women specialist and like that. To also include such scenarios in the account while doing simulation we can create multiple class for customers. We can do that as –

```
import ciw
>>> N = ciw.create_network(
...     arrival_distributions={'Class 0':
[ciw.dists.Exponential(rate=1.0),
...                                       ciw.dists.NoArrivals(),
...                                       ciw.dists.NoArrivals()],
...                            'Class 1':
[ciw.dists.Exponential(rate=2.0),
...                                       ciw.dists.NoArrivals(),
...                                       ciw.dists.NoArrivals()]},
...     service_distributions={'Class 0':
[ciw.dists.Exponential(rate=4.0),
...
ciw.dists.Exponential(rate=1.0),
...
ciw.dists.Deterministic(value=0.0)],
...                            'Class 1':
[ciw.dists.Exponential(rate=6.0),
...
ciw.dists.Deterministic(value=0.0),
...
ciw.dists.Exponential(rate=1.0)]},
...     routing={'Class 0': [[0.0, 1.0, 0.0],
...                          [0.0, 0.0, 0.0],
...                          [0.0, 0.0, 0.0]],
...              'Class 1': [[0.0, 0.0, 1.0],
...                          [0.0, 0.0, 0.0],
...                          [0.0, 0.0, 0.0]]},
```

```
...     number_of_servers=[1, 2, 3],
... )
```