Multiple Server Queue with Reneging Customers

# Description

Impatient customers arrive to a multiple server system, waiting in a single queue if no server is available. Each customer has a certain “impatience” index – an amount of time they are willing to wait in line before being served. If a customer’s time elapses while still in queue, that customer will exit the queue (“renege”).

# Event Graph Model

This model illustrates the following features of Event Graphs:

* Passing arguments to events on a scheduling edge
* Cancelling an event
* Using a container as a state variable

## Parameters

* {tA} – customer interarrival times
* {tS} – customer service times
* k = number servers
* {tR} – “renege” times (time a given customer is willing to wait in line)

## State Variables

* S = number of available servers (initially k)
* q = fifo container of (unique) customer id numbers (initially empty)
* N = number of customer arrivals (initially 0)
* R = number of reneges (initially 0)

## Event Graph



Figure . Multiple Server Queue with Reneging Customers Event Graph

### Notes About the Event Graph in Figure 1

* The state variable q is a first-in first-out (fifo) container except when accessed by the Renege event. There, the particular customer who reneged is removed
* Every arriving customer schedules a Renege event, even if there is a server available.
* A cancelling edge that cancels an event with a parameter, such as the Start-Renege(c) edge, will remove the Renege(c) event whose argument c is exactly equal to the value on the edge, in this case “d,” the id of the customer who was removed from the queue to start service.
* Customers will not, in general, renege in the same order as they arrive. For example, an earlier customer with a long renege time will be in the queue after a subsequent customer with a shorter renege time has left.

# DESpy Implementation

The DESpy implementation of the Event Graph in Figure 1 illustrates the following features of DESpy:

* The state variable q will be represented by a list, and the push/pop calls will use heappush() and heappop() in the (Python built-in) heapq package. The call to remove(c) in the Renege(c) event is a method of Python’s list
* Scheduling an event with an argument is done by a call to schedule() with the additional argument last in the list. There can be as many as needed, but there must be a match between the schedule and the event’s signature.
* Canceling an event is done by a call to cancel(). The first argument is the name of the event, and optional additional arguments are to match an event with arguments.

# Extension: Estimate Delays in Queue and Time in System

Suppose we wish to also estimate the delays in queue for customers served and customers who reneged, as well as the time in the system for those who were served. We can do this by extending the model in Figure 1 to include state variables for these three values; the question then becomes how to obtain them.

Customers entering the system occur at the Enter event, so if we save the time of arrival for each customer, then we can compute the respective values at their corresponding events. Namely, delay in queue for those served at the Start event, delay in queue for those who renege at the Renege(c) event, and the time in the system for those served.

The approach will be to modify the model in Figure 1 as follows:

## Additional State Variables

* a = map of times each customer enters the system (initially empty)
* DS = delay in queue for customers who received service (initially undefined)
* DR = delay in queue for customers who reneged (initially undefined)
* W = time in system for customers who received service (initially undefined)

## New Event Graph



Figure . Event Graph to Tally Delays and Time in System

The state variables DS, DR, and W are *not* time-persistent, as the ones we have been using so far (such as Q and S). They only “exist” for an instant in time. Therefore, they are not initialized in the Run event, and are implemented as local variables in the event methods.

The Leave event now has an argument, which is the customer id of the one completing service. Note that the map a has the corresponding element removed. Similarly, the Renege event also removes the arrival time for the reneging customer.