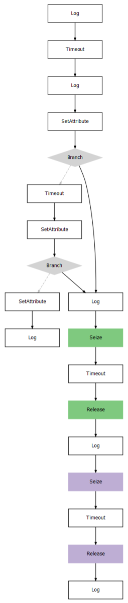
Coding the simulation



Outline

- Student trajectory
- Visualising trajectory
- A single replication
- Multiple replications

Learning Objectives

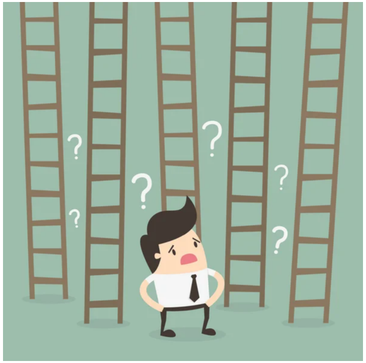
1. Learn the code for the student trajectory.
2. Learn to run multiple replications of the model.

Libraries used

- In order to run the code in this video, you will need to load the following libraries in RStudio:

```
library(simmer)
library(simmer.plot)
library(triangle)
```

Student trajectory



Student trajectory

overview

- The following code is truncated to show an overview.

```
student <-
  trajectory("Student") %>%
  log_("Here I am") %>%
  ...
  seize("MalWart") %>%
  ...
  release("MalWart") %>%
  ...
  seize("JHBunt") %>%
  ...
  release("JHBunt") %>%
  log_(function() {paste("I am leaving!: ", now(mixer))})
```

Student trajectory

cont'd

- Focus on the top chunk of the student trajectory.

```
student <-
  trajectory("Student") %>%
  log_("Here I am") %>%
  timeout(function() {runif(1, min=20/60, max=45/60)}) %>%
  log_("I have affixed my name tag!") %>%
  set_attribute("type",1) %>%
  branch(function() runif(1) < 0.5, continue = TRUE,
    trajectory() %>%
      timeout(function() {rtriangle(1, 10, 45, 15)}) %>%
      set_attribute("type",2) %>%
      branch((function() runif(1) < 0.1), continue = FALSE,
             trajectory() %>%
             set_attribute("type",3) %>%
             log_(function() {paste("I am timid/tired...
                                    leaving!: ", now(mixer))}))
         ) %>%
```

Student trajectory

cont'd

- Focus on the bottom chunk of the student trajectory.

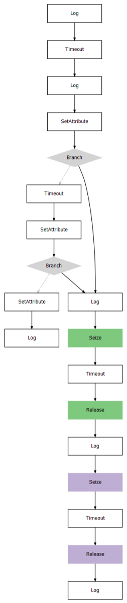
```
...
log_("I shall join the queue and talk to a MalWart
      recruiter") %>%
seize("MalWart") %>%
timeout(function() rexp(1, 1/3)) %>%
release("MalWart") %>%
log_("I am done chatting with MalWart recruiter, now joining
      the queue and talk to a JHBunt recruiter") %>%
seize("JHBunt") %>%
timeout(function() rexp(1, 1/6)) %>%
release("JHBunt") %>%
log_(function() {paste("I am leaving!: ", now(mixer))})
```

Visualising trajectory

- It is easy to make a mistake when coding up complicated trajectories.
- One useful tool for checking our implementation is a visualisation of the trajectory.
- In the flowchart, each resource will be coded with a different colour, while branch conditions will be coloured grey.

```
get_palette <- brewer_pal(type = "qual",
                           palette = 1)
plot(student, fill = get_palette)
```

  - Take note that to export the image to a file, we need a few extra lines of code.



A single replication

- Simulation environment:

```
mixer <-
  simmer("mixer") %>%
  add_resource("MalWart", capacity = 2) %>%
  add_resource("JHBunt", capacity = 3) %>%
  add_generator("Student", student, function() rexp(1, 1/2),
                mon = 2)

mixer %>% run(until = 60)
```

Multiple replications

- A single replication provides us with summary performance measures.
- But in order to rely on the Central Limit Theorem when computing confidence intervals, we need to have replications.
- As we saw from earlier weeks, we can use the function `replicate` to do this, but we must remember to call "wrap" at the end of the statements.

```
mm1.envs  <- replicate(10,
  simmer("mixer") %>%
  add_resource("MalWart", capacity = 2) %>%
  add_resource("JHBunt", capacity = 3) %>%
  add_generator("Student", student, function() rexp(1, 1/2),
                  mon = 2) %>%
    run(until = 360) %>% wrap()
  )
```

Summary

In this video, we have:

1. Learned the code for the student trajectory.
2. Learned to run multiple replications of the model.