INSTITUTO UNIVERSITÁRIO DE LISBOA

Modelação Estocástica - Trabalho 3

Ciência de Dados - PL - 3º ano| Professora: Catarina Marques

Catarina Castanheira, 92478 João Martins, 93259 Joel Paula, 93392

14/11/2021

```r
knitr::opts_chunk$set(echo = TRUE)
# Package que contém geradores para a Distribuição Triangular:
library(extraDistr)
# ver https://search.r-project.org/CRAN/refmans/extraDistr/html/Triangular.html
library(simmer)
library(simmer.plot)
```

Simulação do Check-in de uma Companhia Aérea

Condições:

- Os voos são de 170 passageiros, contudo a ocupação dos voos pode não ser completa;
- O Check-in abre 2h antes da partida do voo e fecha 40 min antes da hora da partida;
- Chegada de passageiros – os passageiros chegam ao *check-in* de acordo com uma distribuição de Poisson com média diferente consoante o tipo de passageiro:
  - *Economy* (média - 1,7);
  - *Business* (média - 0,85);
- Os passageiros *Business*, assim como os passageiros que sejam membros *Gold*/*Premium* são 20% do total dos passageiros. São considerados no sistema como passageiros *Vip* e são servidos por apenas um balcão de passageiros – o *balcão Vip*;
- Os passageiros *Vip* fazem o *check-in* entre os 80 e 40 minutos antes do voo;
- Os passageiros podem fazer o *check-in* individualmente ou em grupo; no entanto, por uma questão de simplicidade, vamos considerar que os grupos de passageiros fazem o *check-in* individualmente, pelo que são considerados como passageiros individuais;
- A empresa de viação disponibiliza dois balcões para passageiros *Economy* e um para passageiros *Vip*. Existe ainda uma máquina que pode ser utilizada por passageiros *Economy* que não tenham que despachar bagagem;
- A fila do *check-in* para os passageiros *Economy* que tenham bagagem para despachar pode ser única ou por balcão;
- 75% dos passageiros *Economy* despacham a bagagem. Neste caso, o tempo de atendimento no *check-in* demora mais 30 segundos;
- O tempo de atendimento no *check-in* (tempo de serviço) tem distribuição triangular com parâmetros apresentados na tabela seguinte (em segundos):

	Min	Moda	Max
Balcão	60	103	120
Máquina	40	60	90

- Os voos podem ser servidos por mais de um balcão; Do mesmo modo, os balcões podem servir um único voo ou vários voos. Contudo, iremos considerar que apenas estamos a servir um voo.

Analise o tempo total do processo de check-in nos dois cenários seguintes:

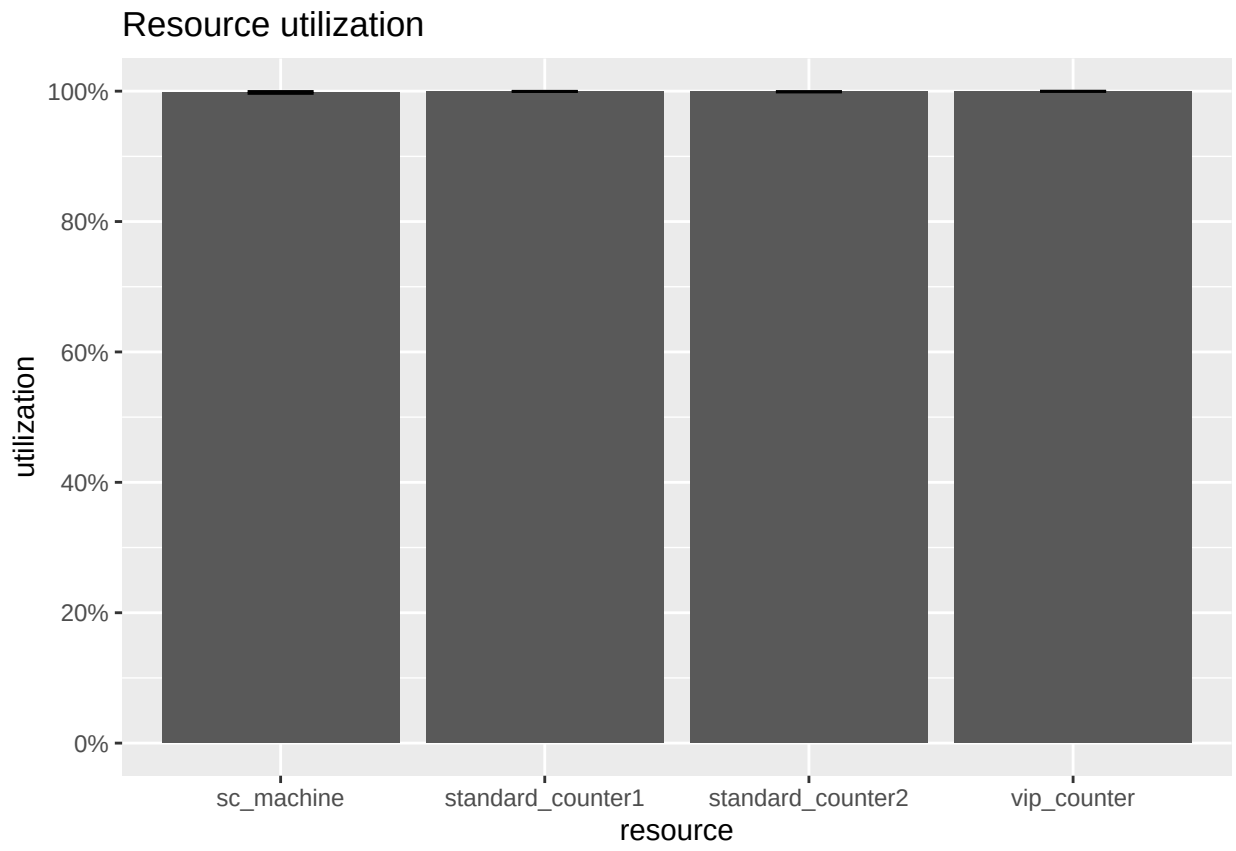
1) Análise do impacto da estratégia de fila dos balcões não *VIP* na distribuição do tempo total do *check-in*

> Nota: decidimos converter todos os tempos em segundos, já que achamos que seria melhor ter essa resolução, para cumprir todas as premissas do problema.

```
runs <- 100
# Notas:
# Se checkin esta aberto entre os 120min (2h00) e os 40min antes do voo, significa estar aberto
# durante 80min (4800s).
# Se checkin para vip esta aberto entre os 80min e os 40min antes do voo, significa estar
# aberto durante 40min (2400s)

# definicao de tempos (em segundos) e outros parametros necessarios:
overall_checkin_time <- 4800 # tempo total do processo de checkin (80min = 4800s)
close_checkin_all <- overall_checkin_time # t_i em que fecha o checkin para todos os passageiros
begin_checkin_economy <- 1 # t_i em que abre o checkin para passageiros economy
begin_checkin_vip <- 2400 # t_i em que abre o checkin para passageiros vip (40min = 2400s)
dur_checkin_vip <- begin_checkin_vip
dur_checkin_economy <- overall_checkin_time

triang_counter_min <- 60 # parametro para distribuicao triangular do atendimento do checkin
triang_counter_mode <- 103 # parametro para distribuicao triangular do atendimento do checkin
triang_counter_max <- 120 # parametro para distribuicao triangular do atendimento do checkin

triang_machine_min <- 40 # parametro para distribuicao triangular do atendimento do checkin
triang_machine_mode <- 60 # parametro para distribuicao triangular do atendimento do checkin
triang_machine_max <- 90 # parametro para distribuicao triangular do atendimento do checkin

vip_passenger_lambda <- 0.85 # parametro lambda para gerador Poisson de passageiros VIP
economy_passenger_lambda <- 1.7 # parametro lambda para gerador Poisson de passageiros Economy

extra_time_luggage <- 30 # tempo de atendimento extra no caso de passageiro economy ter bagagem

n_max <- 170 # maximo de passageiros por voo
prop_economy <- 0.8 # proporcao de passageiros economy
n_max_economy <- n_max * prop_economy # maximo de passageiros da class economy por voo
n_max_vip <- n_max - n_max_economy # maximo de passageiros vip por voo

# Definicao da trajectoria dos passageiros VIP
vip_passenger <-
    trajectory("VIP passenger's path") %>%
    # log_("Arrived at Check-In Area.") %>%
    seize("vip_counter") %>%
    # log_("Making Check-In...") %>%
    timeout(function() rtriang(
        n = 1,
        a = triang_counter_min,
        b = triang_counter_max,
        c = triang_counter_mode)) %>% # a=min, b=max, c=moda
    release("vip_counter") #%>%
    #log_("Check-in completed! Leaving VIP Counter and going to Security Checkpoint.")

# Definicao da trajectoria dos passageiros Economy

economy_passenger <-
```

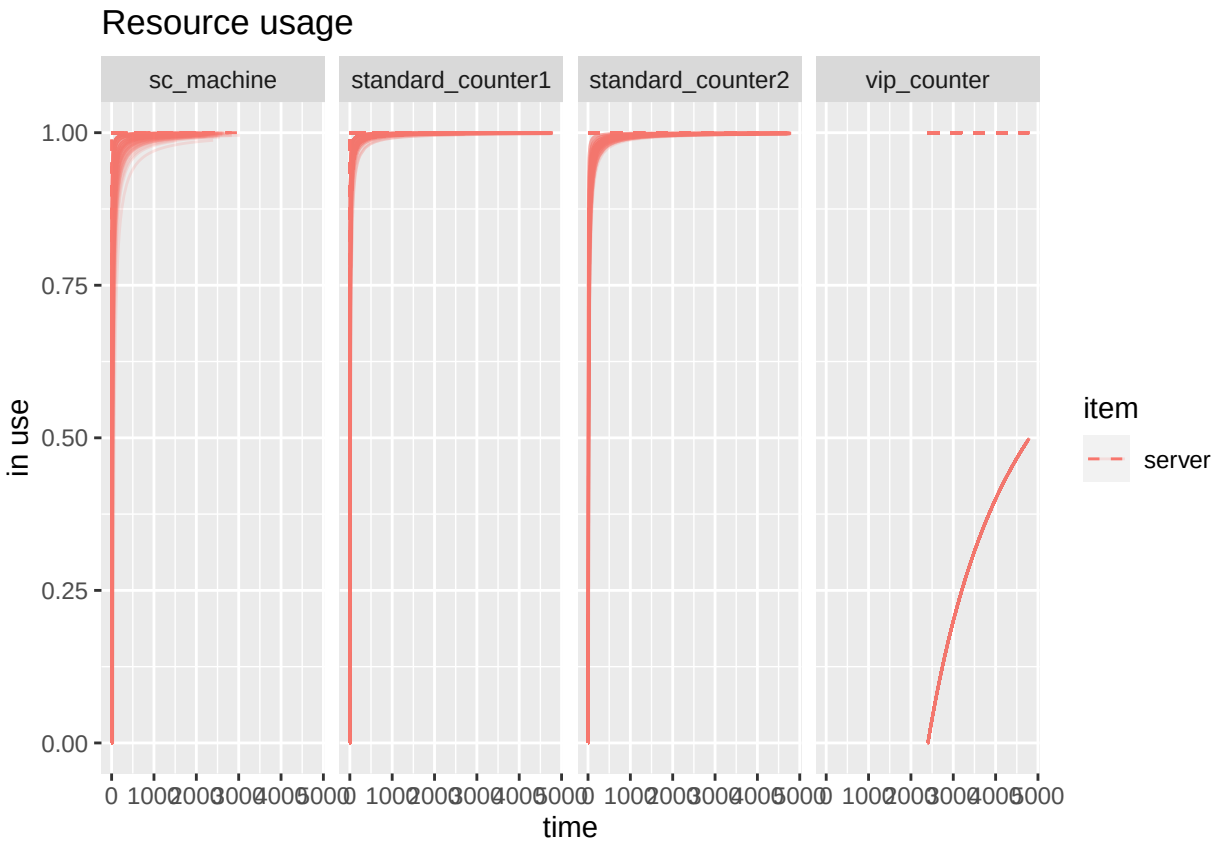
```
    trajectory() %>%
    #log_("Arrived at Check-In Area.") %>%
    branch(
        # 75% tem bagagem, 25% nao tem bagagem
        function() (runif(1) > 0.75) + 1, continue = c(FALSE, FALSE),
        trajectory() %>% # trajectoria do passageiro se tiver bagagem
            # log_("I have luggage to check.") %>%
            # cada balcao com fila individual
            select(c("standard_counter1", "standard_counter2"), policy = "shortest-queue") %>%
            seize_selected() %>%
            # log_("Making Check-In...") %>%
            timeout(function() rtriang(
                # a=min, b=max, c=moda; tem tempo extra da bagagem
                n = 1,
                a = triang_counter_min,
                b = triang_counter_max,
                c = triang_counter_mode) + extra_time_luggage) %>%
            release_selected() , #%>%
            # log_("Check-in completed! Leaving Standard Counter, to the Security Checkpoint."),
        trajectory() %>% # trajectoria do passageiro sem bagagem
            # log_("I just have carry-on.") %>%
            seize("sc_machine") %>%
            # log_("Making Check-In...") %>%
            timeout(function() rtriang(
                n = 1,
                a = triang_machine_min,
                b = triang_machine_max,
                c = triang_machine_mode)) %>% # a=min, b=max, c=moda
            release("sc_machine") #%>%
            # log_("Check-in completed! Leaving Self Check-In, to the Security Checkpoint.")
    )

# horarios de funcionamento do checkin para passageiros VIP
vip_checkin_schedule <-
    schedule(
        c(begin_checkin_vip, close_checkin_all),
        # capacidade de atender 1 passageiro de cada vez
        c(1, 0), period = overall_checkin_time)

# horarios de funcionamento do checkin para passageiros Economy
    # no nosso caso concreto nao seria necessario definir porque:
        # tempo checkin para economy = tempo total em que checkin esta aberto;
    # deixamos aqui definido para poder ser manipulado
economy_checkin_schedule <-
    schedule(
        c(begin_checkin_economy, close_checkin_all),
        # capacidade de atender 1 passageiro de cada vez
        c(1, 0), period = overall_checkin_time)


set.seed(42) # Simulation seed

# Ambiente de Simulacao (100 vezes)
envs <- lapply(1:runs, function(i) {
    simmer("Check-In") %>%
    add_resource("vip_counter", capacity = vip_checkin_schedule) %>%
```

```
      # so 1 balcao vip, sujeito ao horario checkin vip
    add_resource("standard_counter1", capacity = economy_checkin_schedule) %>%
      # 2 balcoes para passageiros economy
    add_resource("standard_counter2", capacity = economy_checkin_schedule) %>%
    add_resource("sc_machine", capacity = economy_checkin_schedule) %>%
      # temos uma maquina self-checkin para passageiros sem bagagem
    add_generator("VIP Passenger",
      # "gerador" de passageiros VIP chegam ao check-in com uma distribuicao Poisson (0.85)
                  vip_passenger,
                  from_to(start_time = begin_checkin_vip,
                          stop_time = close_checkin_all,
                          dist = function() {
                            c(rpois(n = n_max_vip, lambda = vip_passenger_lambda), -1)
                            },
                          arrive = F),)  %>%
    add_generator("Economy Passenger",
      # "gerador" de passageiros Economy chegam ao check-in com uma distribuicao Poisson (1.7)
                  economy_passenger,
                  function() {
                    c(rpois(n = n_max_economy, lambda = economy_passenger_lambda), -1)
                    }) %>%
    run(until = close_checkin_all) # Simulador do check-in (desenrola-se por 1h20)
})

arrivals <- get_mon_arrivals(envs)
resources <- get_mon_resources(envs)

plot(resources, metric = "utilization")
```



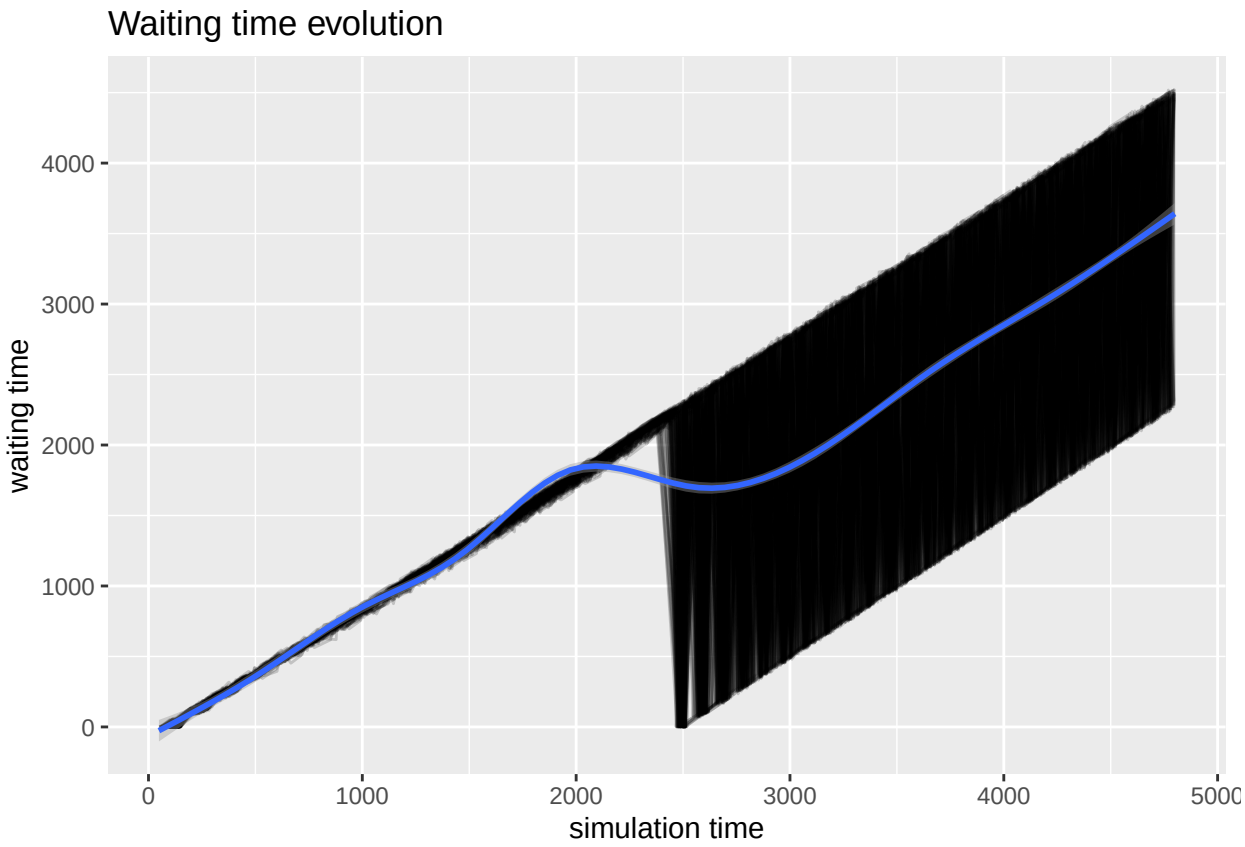
```
plot(resources, metric = "usage",item = "server")
```

```
## Warning: Removed 2 row(s) containing missing values (geom_path).
```

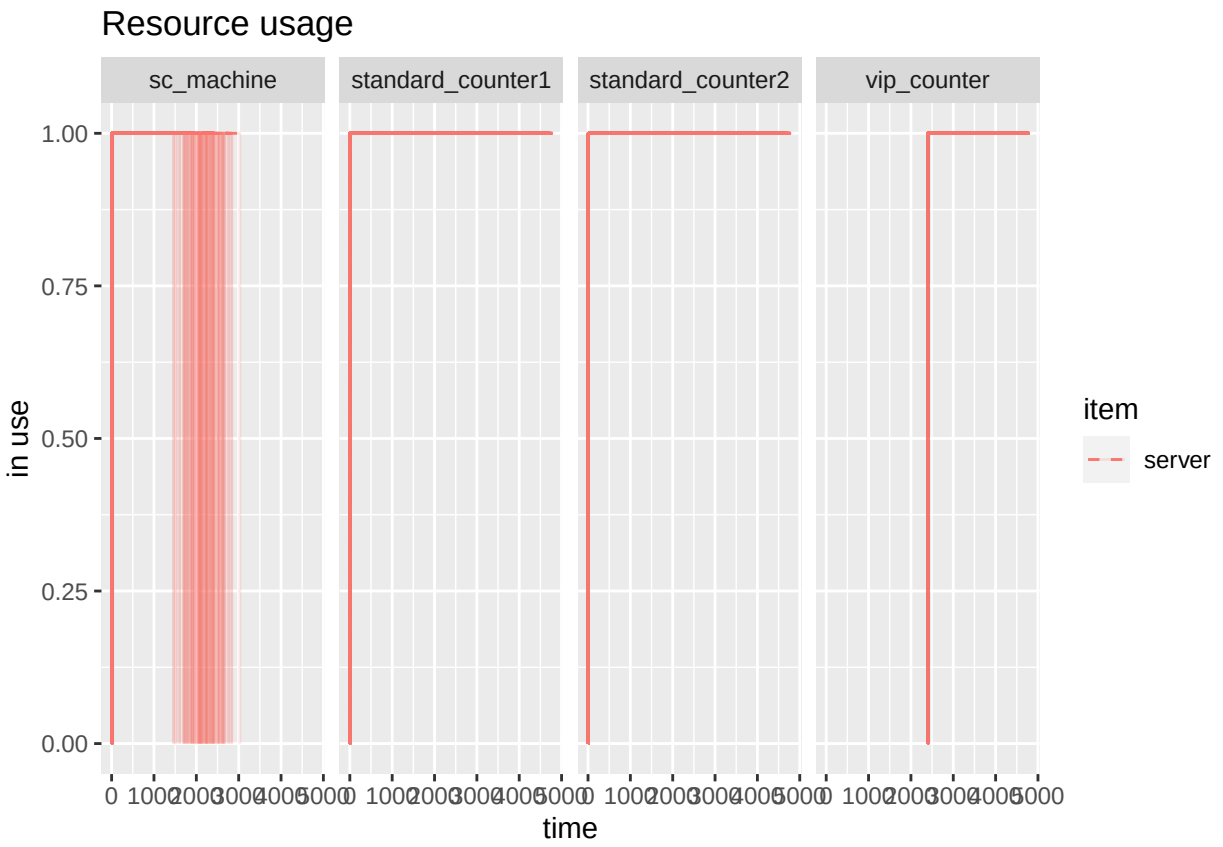



```
plot(arrivals, metric="waiting_time")
```

```
## `geom_smooth()` using method = 'gam' and formula 'y ~ s(x, bs = "cs")'
```

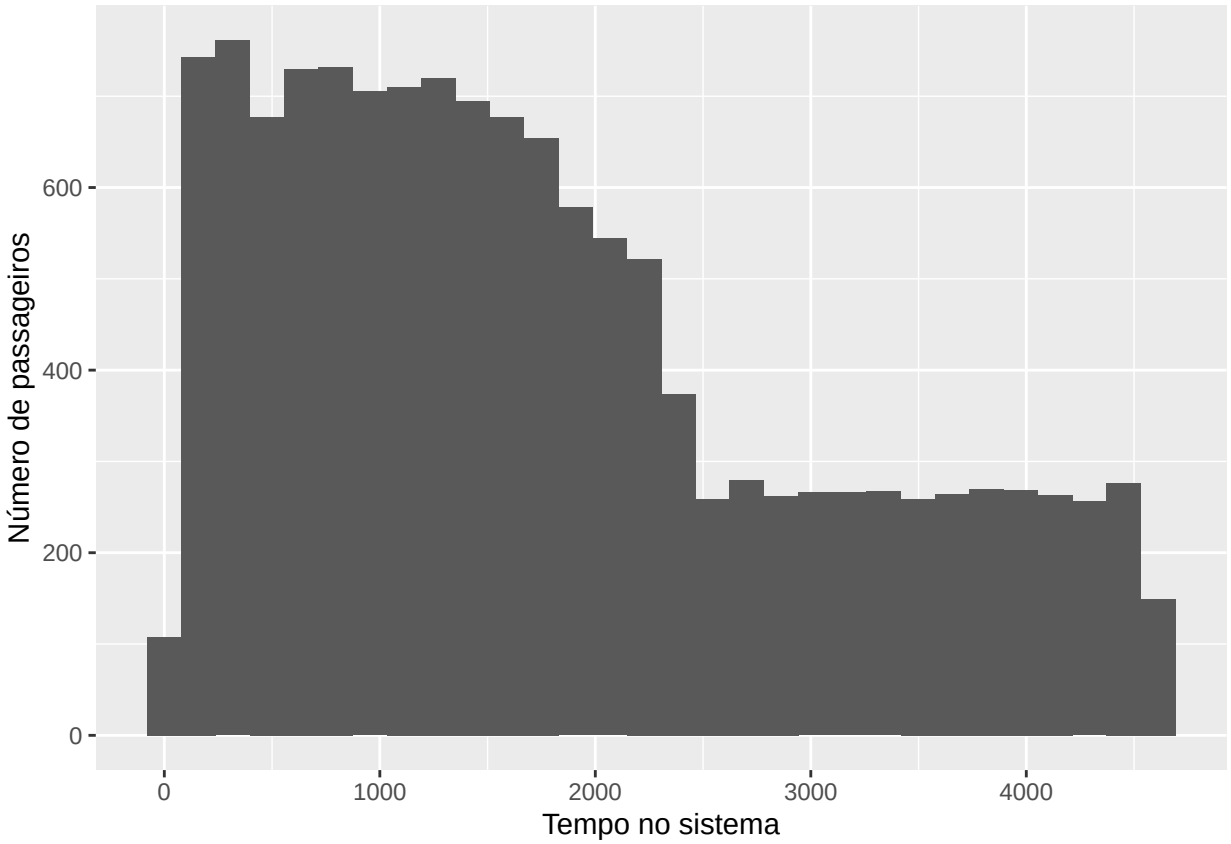



```
plot(resources, metric = "usage", item = "server", steps = TRUE)
```




```
arrivals %>%
  ggplot(aes(end_time - start_time)) +
  geom_histogram() +
  xlab("Tempo no sistema") +
  ylab("Número de passageiros")
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```




```
# número de passageiros que não conseguiram terminar o check-in
(avg_left_behind_passengers <- nrow(arrivals[!arrivals$finished,])/runs)
```

```
## [1] 0
```

```
# Cálculo do tempo de espera
arrivals <- transform(arrivals, waiting_time = end_time - start_time - activity_time)
#tempo de espera em minutos
summary(arrivals$waiting_time/60)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    0.00   11.72   24.28   28.29   41.22   75.37
```

```
#tempo de espera em minutos dos VIP
summary(arrivals[grepl("VIP Passenger", arrivals$name),]$waiting_time/60)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   0.000   8.899  18.670  18.582  28.359  38.488
```

Nesta primeira simulação, em que temos filas individuais para cada balcão de atendimento, verificamos o seguinte:

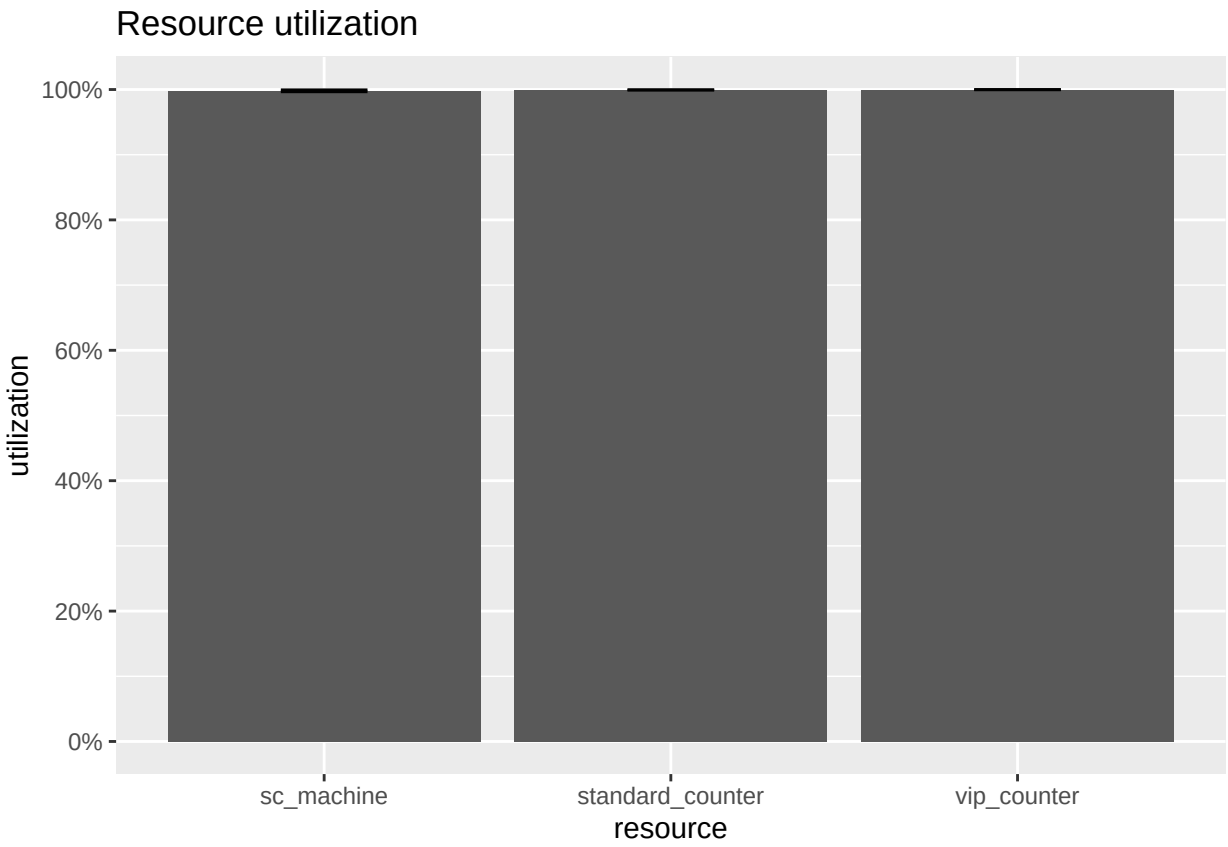
- Qualquer um dos recursos (máquina de *check-in*, balcão *VIP*, e os dois balcões para classe *Economy*) são usados a 100% (gráficos *Resource Utilization* e *Resource Usage*), a partir do momento em que aparecem passageiros. Nos gráficos respeitantes ao *"Resource Usage"* com o parâmetro *"steps"* activo, podemos verificar que a máquina de *check-in* tem diversos momentos em que não é utilizada.

- O tempo de espera médio segue uma tendência linearmente crescente (veja-se o gráfico *"Waiting Time Evolution"*); contudo, quando começam a aparecer os passageiros *VIP*, o tempo de espera médio sofre uma quebra face ao tempo total da simulação, mantendo a tendência crescente. Isto deve-se ao facto de esses passageiros estarem a chegar naquele momento, tendo um tempo de espera ainda curto e serem imediatamente atendidos.
- Relativamente aos tempos, verificamos que a média de espera se situa nos 28.29 minutos, tendo como máximo 75.37 minutos. Se considerarmos somente os passageiros *VIP*, a média cai para os 18.58 minutos e o máximo para os 38.48 minutos.

Experimentando a fila única para os passageiros Economy:

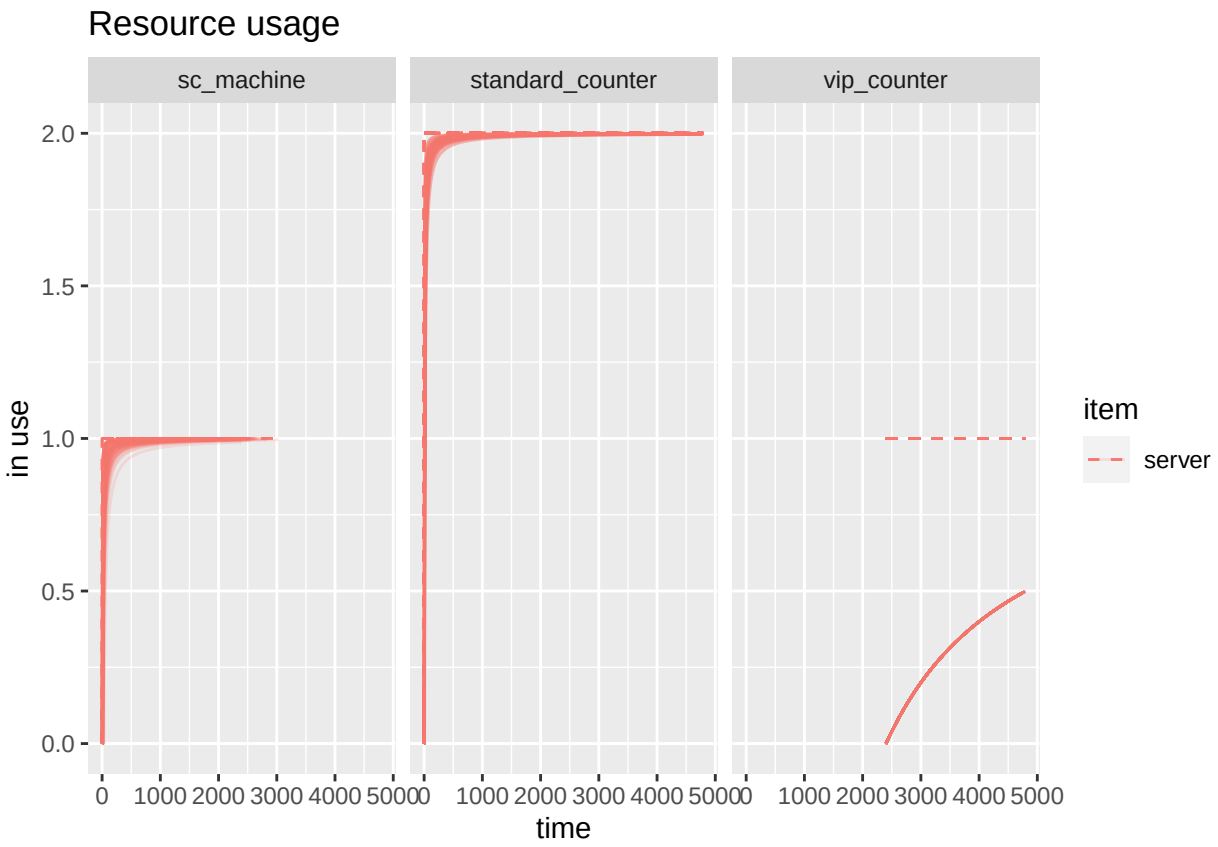
```
# Definicao da trajectoria dos passageiros Economy fila única

economy_passenger <-
    trajectory() %>%
    #log_("Arrived at Check-In Area.") %>%
    branch(
        # 75% tem bagagem, 25% nao tem bagagem
        function() (runif(1) > 0.75) + 1, continue = c(FALSE, FALSE),
        trajectory() %>% # trajectoria do passageiro se tiver bagagem
            #log_("I have luggage to check.") %>%
            seize("standard_counter") %>%
            #log_("Making Check-In...") %>%
            timeout(function() rtriang(
                n = 1,
                a = triang_counter_min,
                b = triang_counter_max,
                c = triang_counter_mode) + extra_time_luggage) %>%
              # a=min, b=max, c=moda; tem tempo extra da bagagem
            release("standard_counter"), # %>%
            #log_("Check-in completed! Leaving Standard Counter to the Security Checkpoint."),
        trajectory() %>% # trajectoria do passageiro sem bagagem
            #log_("I just have carry-on.") %>%
            seize("sc_machine") %>%
            #log_("Making Check-In...") %>%
            timeout(function() rtriang(
                n = 1,
                a = triang_machine_min,
                b = triang_machine_max,
                c = triang_machine_mode)) %>% # a=min, b=max, c=moda
            release("sc_machine") #%>%
            #log_("Check-in completed! Leaving Self Check-In to the Security Checkpoint.")
    )

# horarios de funcionamento do checkin para passageiros Economy
economy_checkin_schedule <-
    schedule(
        c(begin_checkin_economy, close_checkin_all),
        # capacidade de atender 2 passageiros de cada vez
        c(2, 0), period = overall_checkin_time)

# horário da máquina
self_checkin_schedule <-
    schedule(
```

```
        c(begin_checkin_economy, close_checkin_all),
        # capacidade de atender 2 passageiros de cada vez
        c(1, 0), period = overall_checkin_time)

set.seed(42) # repor semente da simulação

# Ambiente de Simulacao (100 vezes)
envs <- lapply(1:runs, function(i) {
    simmer("Check-In") %>%
    add_resource("vip_counter", capacity = vip_checkin_schedule) %>%
      # so 1 balcao vip, sujeito ao horario checkin vip
    add_resource("standard_counter", capacity = economy_checkin_schedule) %>%
      # 2 balcoes para passageiros economy
    add_resource("sc_machine", capacity = self_checkin_schedule) %>%
      # temos uma maquina self-checkin para passageiros sem bagagem
    add_generator("VIP Passenger",
      # "gerador" de passageiros VIP chegam ao check-in com uma distribuicao Poisson (0.85)
                  vip_passenger,
                  from_to(start_time = begin_checkin_vip,
                          stop_time = close_checkin_all,
                          dist = function() {
                            c(rpois(n = n_max_vip, lambda = vip_passenger_lambda), -1)
                            },
                          arrive = F),
                  )  %>%
    add_generator("Economy Passenger",
      # "gerador" de passageiros Economy chegam ao check-in com uma distribuicao Poisson (1.7)
                  economy_passenger,
                  function() {
                    c(rpois(n = n_max_economy, lambda = economy_passenger_lambda), -1)
                    }) %>%
    run(until = close_checkin_all) # Simulador do check-in (desenrola-se por 1h20)
})

arrivals <- get_mon_arrivals(envs)
resources <- get_mon_resources(envs)

plot(resources, metric = "utilization")
```

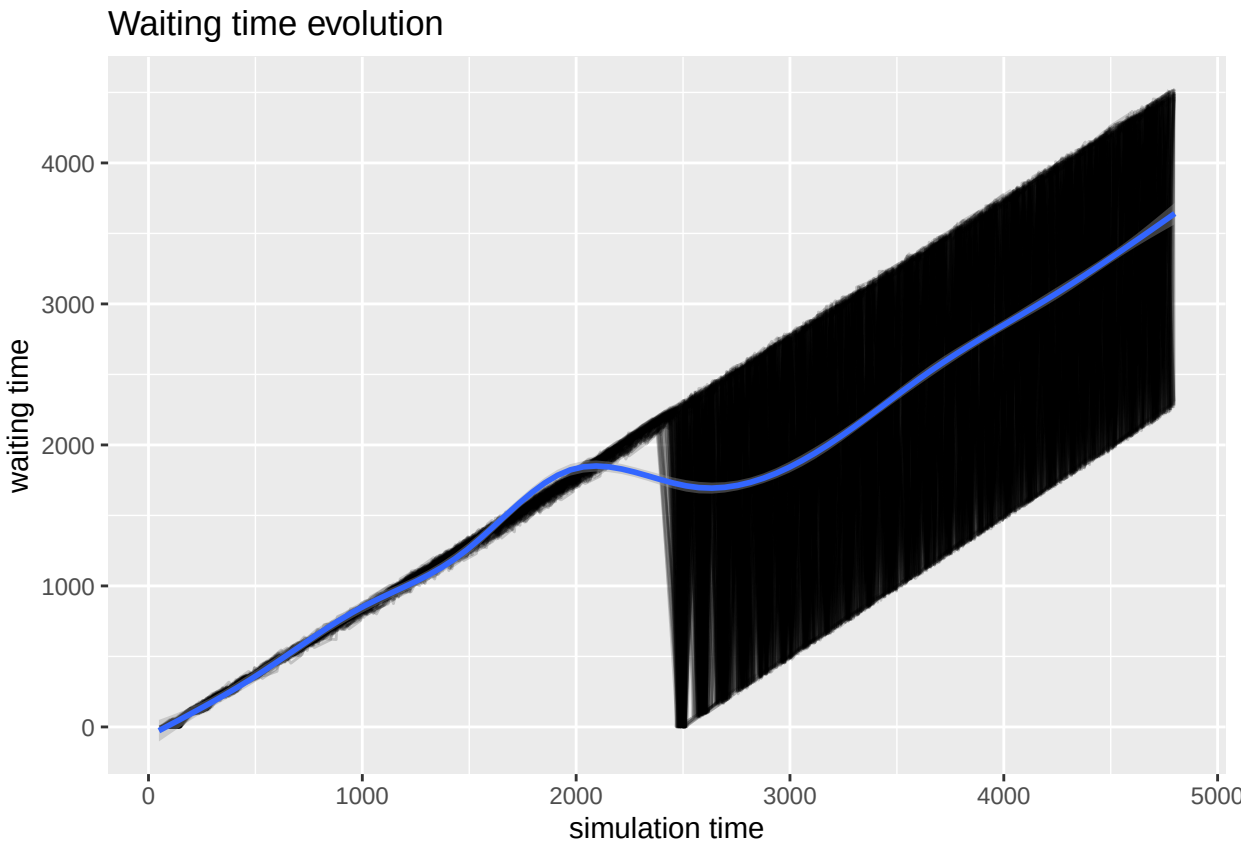



```
plot(resources, metric = "usage",item = "server")
```

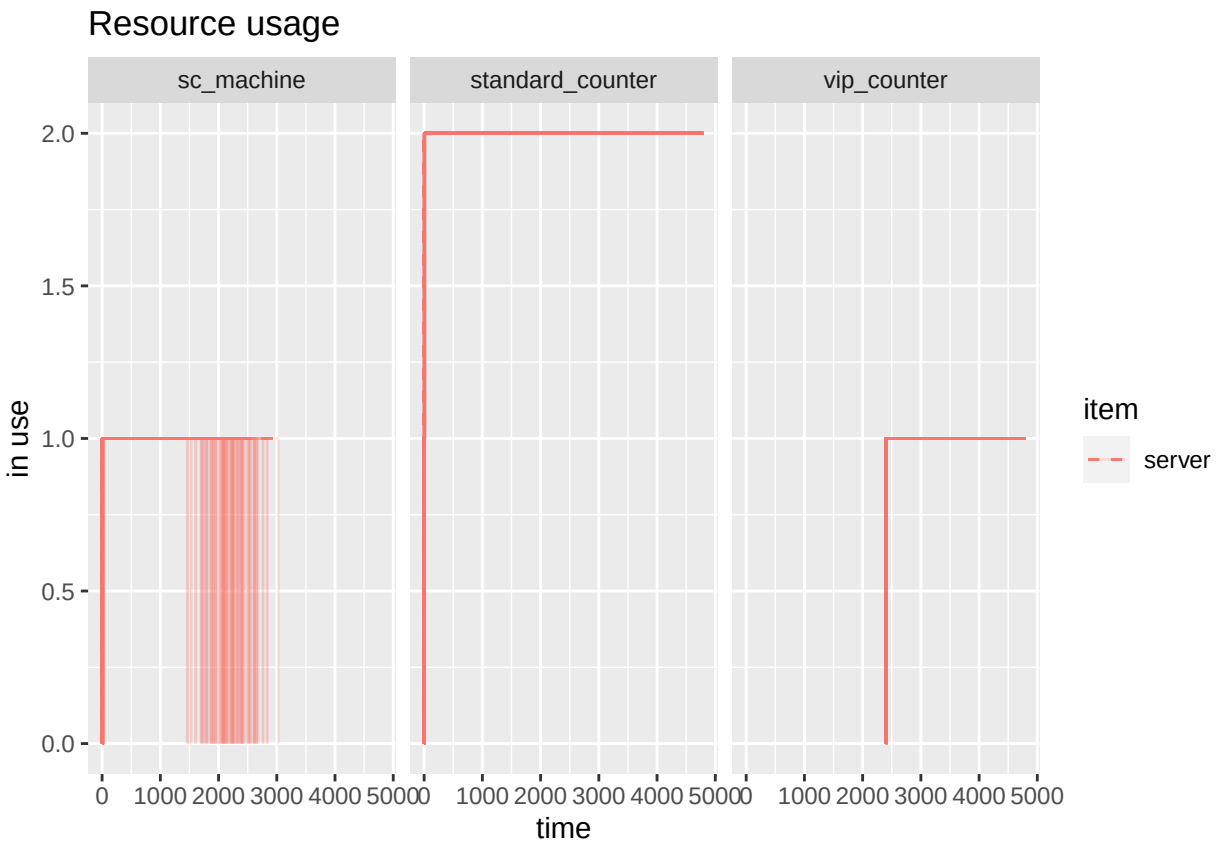
```
## Warning: Removed 2 row(s) containing missing values (geom_path).
```



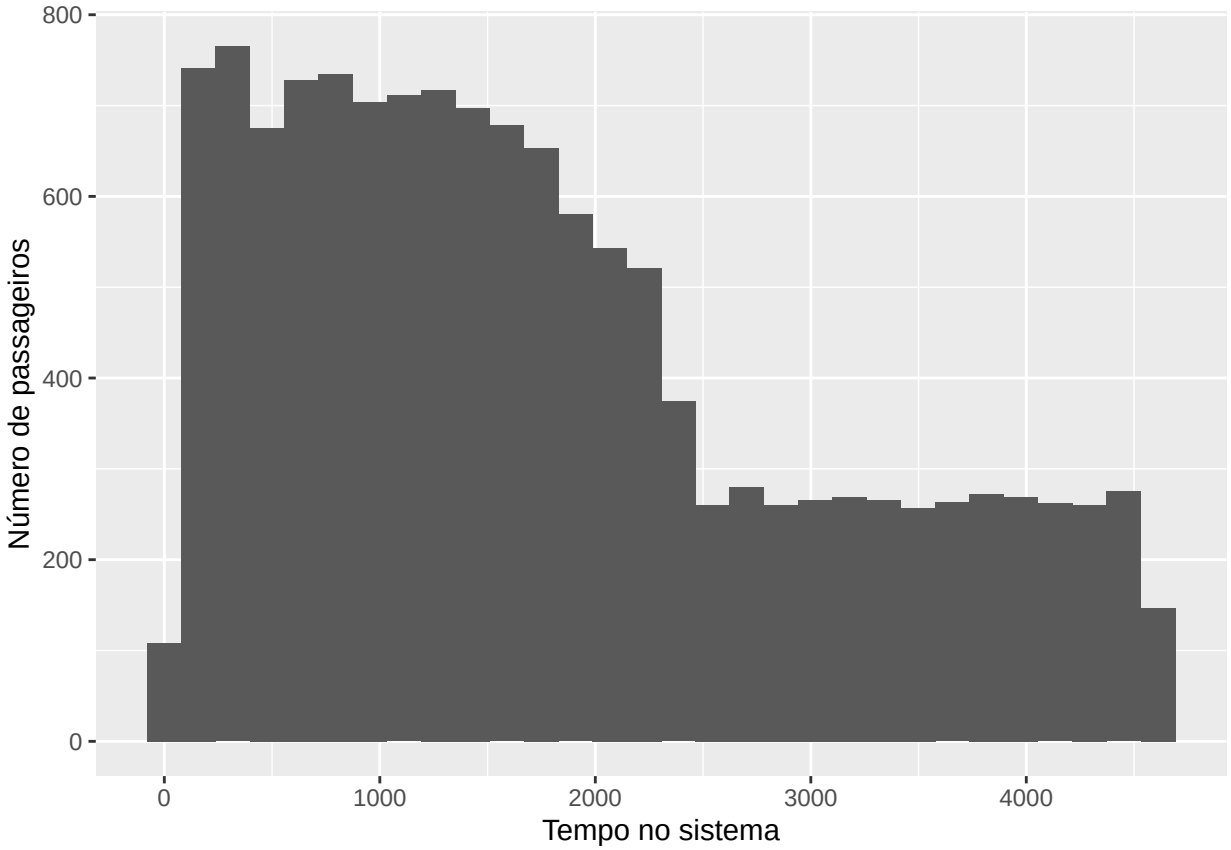

```
plot(arrivals, metric="waiting_time")
```




```
plot(resources, metric = "usage", item = "server", steps = TRUE)
```




```
arrivals %>%
  ggplot(aes(end_time - start_time)) +
  geom_histogram() +
  xlab("Tempo no sistema") +
  ylab("Número de passageiros")
```




```
# número de passageiros que não conseguiram terminar o check-in
(avg_left_behind_passengers <- nrow(arrivals[!arrivals$finished,])/runs)
```

```
## [1] 0
```

```
# Cálculo do tempo de espera
arrivals <- transform(arrivals, waiting_time = end_time - start_time - activity_time)
#tempo de espera em minutos
summary(arrivals$waiting_time/60)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    0.00   11.72   24.29   28.29   41.20   75.34
```

```
#tempo de espera em minutos dos VIP
summary(arrivals[grepl("VIP Passenger", arrivals$name),]$waiting_time/60)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   0.000   8.899  18.670  18.582  28.359  38.488
```

Neste caso em que temos uma fila única de atendimento para os passageiros de classe *Economy*, o que se observa é que não existe alteração significativa dos tempos de espera. Isto revela que dando a qualquer passageiro que chegue de classe *Economy* a possibilidade de escolher ir para a fila do balcão 1 ou 2 que seja mais pequena (como verificado na primeira simulação) ou "obrigando-o" a integrar uma fila única e esperar até que um qualquer balcão esteja disponível para atendimento, tem na prática o mesmo efeito. Qualquer um dos métodos - fila única / uma fila por balcão - poderia ser usado, obtendo-se os mesmos resultados.

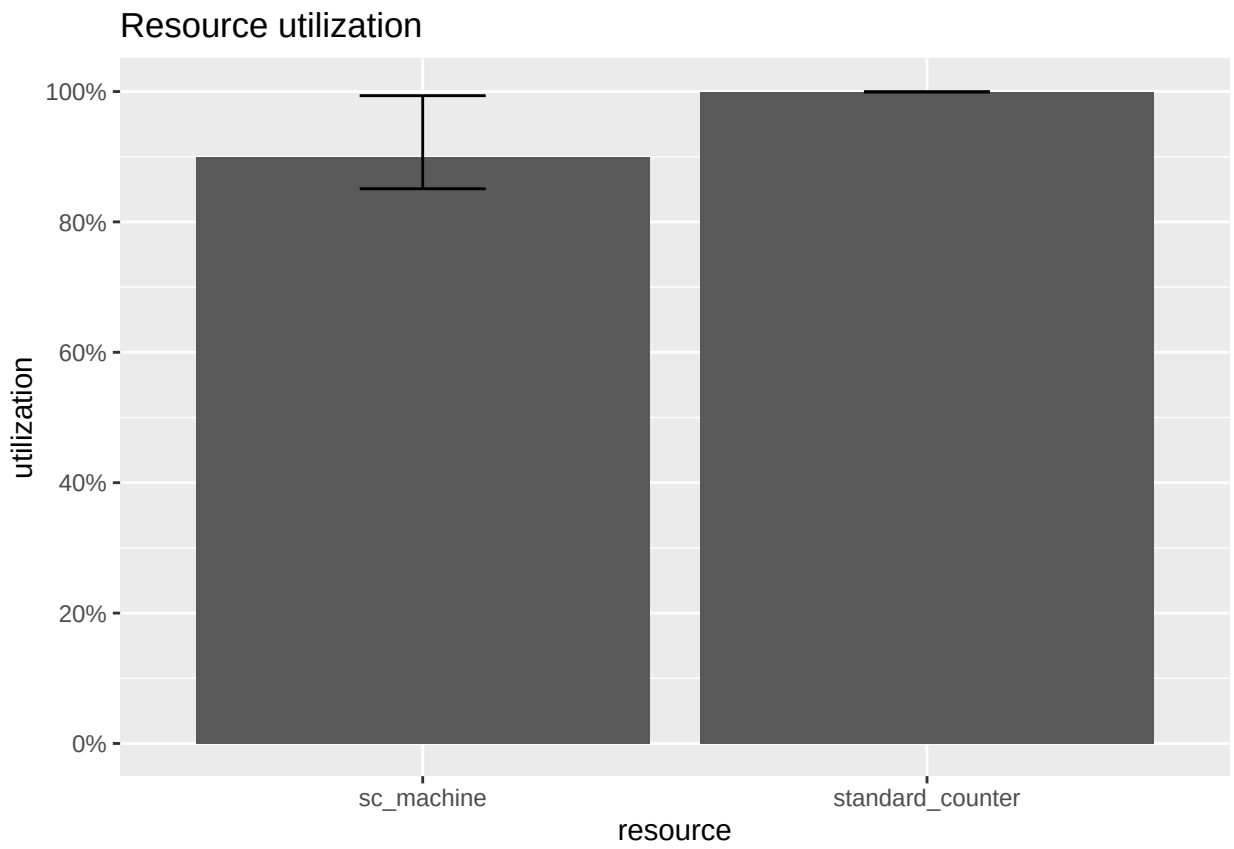
---

2) Análise do impacto da segmentação dos passageiros na distribuição do tempo total do processo.

Ou seja, analise o impacto de os passageiros Vip serem servidos por um balcão que lhes é dedicado ou não existir este balcão e estes passageiros serem servidos pelos outros dois balcões com uma prioridade maior da dos outros passageiros.

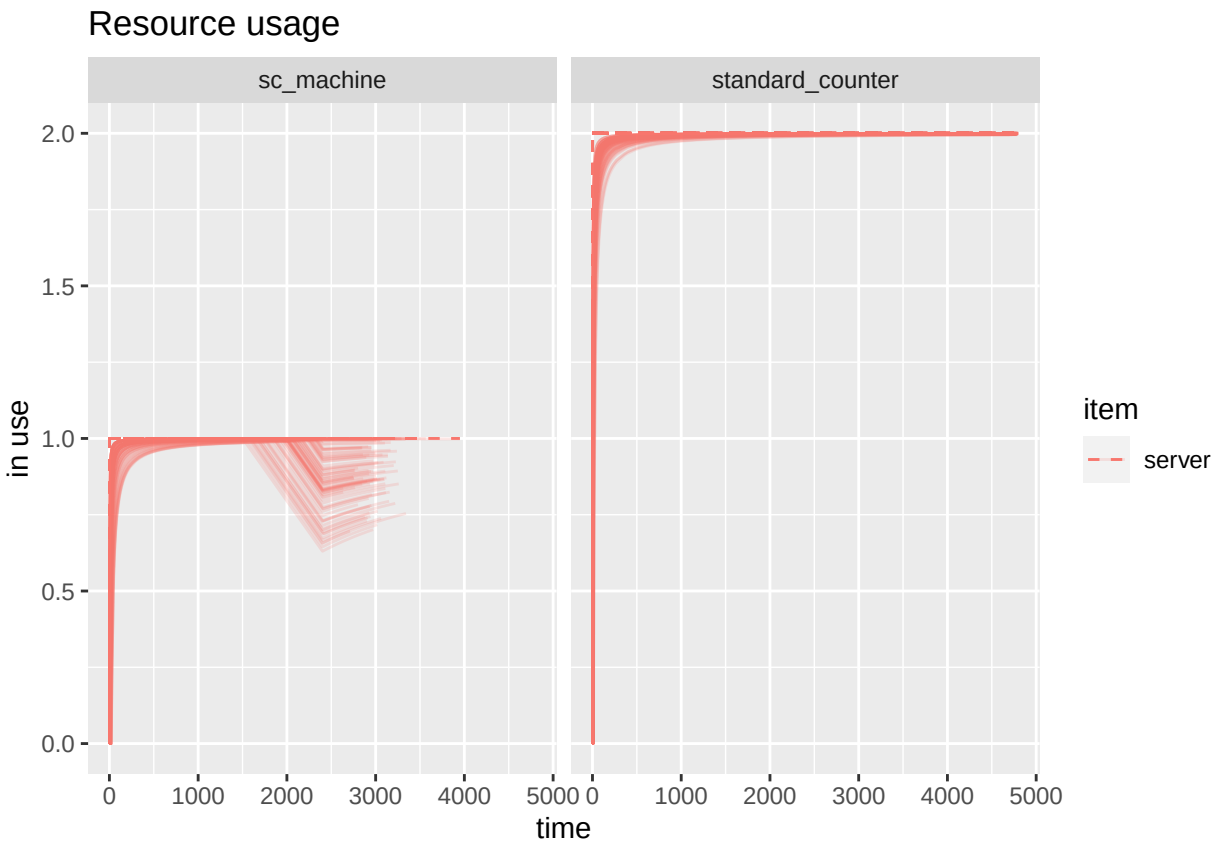
> Nota: para este cenário, vamos assumir que os passageiros VIP têm a mesma proporção de bagagem de porão que os outros passageiros.

```
set.seed(42)

# Ambiente de Simulacao (100 vezes)
envs <- lapply(1:runs, function(i) {
    simmer("Check-In") %>%
    # 2 balcoes para passageiros economy
    add_resource("standard_counter", capacity = economy_checkin_schedule) %>%
    # temos uma maquina self-checkin para passageiros sem bagagem
    add_resource("sc_machine", capacity = self_checkin_schedule) %>%
    add_generator("VIP Passenger",
      # "gerador" de passageiros VIP chegam ao check-in com uma distribuicao Poisson (0.85)
                  economy_passenger,
                  from_to(start_time = begin_checkin_vip,
                          stop_time = close_checkin_all,
                          dist = function() {
                            c(rpois(n = n_max_vip, lambda = vip_passenger_lambda), -1)
                            },
                          arrive = F),
                  # os VIP neste caso seguem a trajetoria dos Economy, mas com maior prioridade
                  priority = 1
                  )  %>%
    add_generator("Economy Passenger",
      # "gerador" de passageiros Economy chegam ao check-in com uma distribuicao Poisson (1.7)
                  economy_passenger,
                  function() {
                    c(rpois(n = n_max_economy, lambda = economy_passenger_lambda), -1)
                    }) %>%
    run(until = close_checkin_all) # Simulador do check-in (desenrola-se por 1h20)
})

arrivals <- get_mon_arrivals(envs)
resources <- get_mon_resources(envs)

plot(resources, metric = "utilization")
```



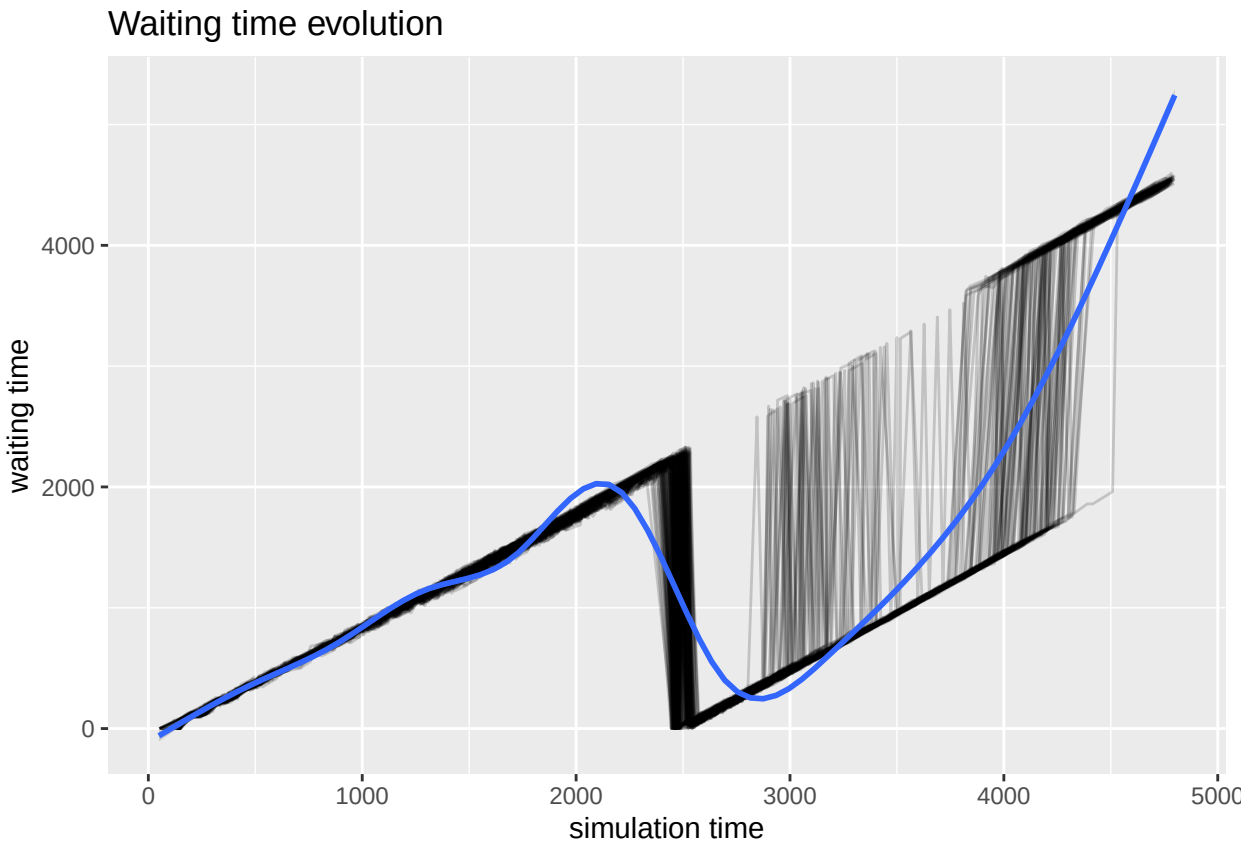

```
plot(resources, metric = "usage",item = "server")
```

```
## Warning: Removed 8 row(s) containing missing values (geom_path).
```

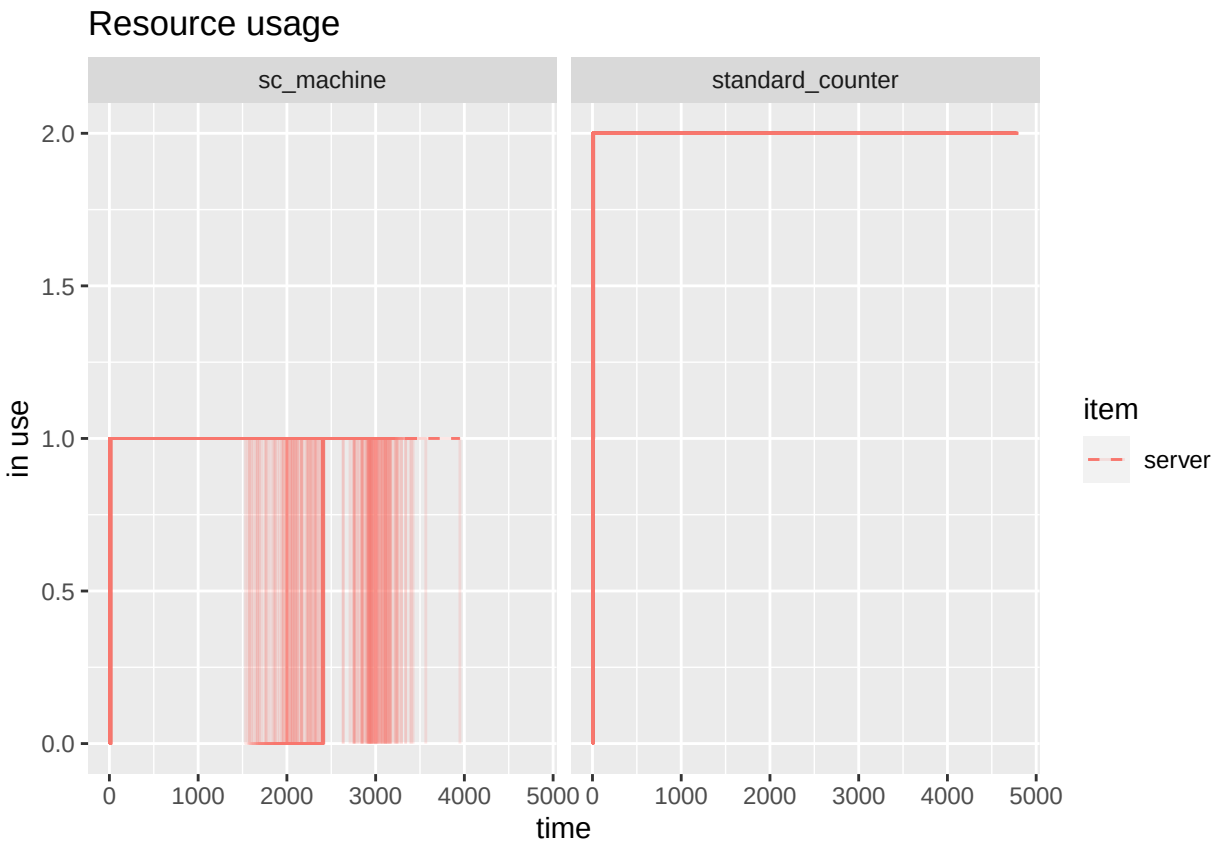



```
plot(arrivals, metric="waiting_time")
```

```
## `geom_smooth()` using method = 'gam' and formula 'y ~ s(x, bs = "cs")'
```

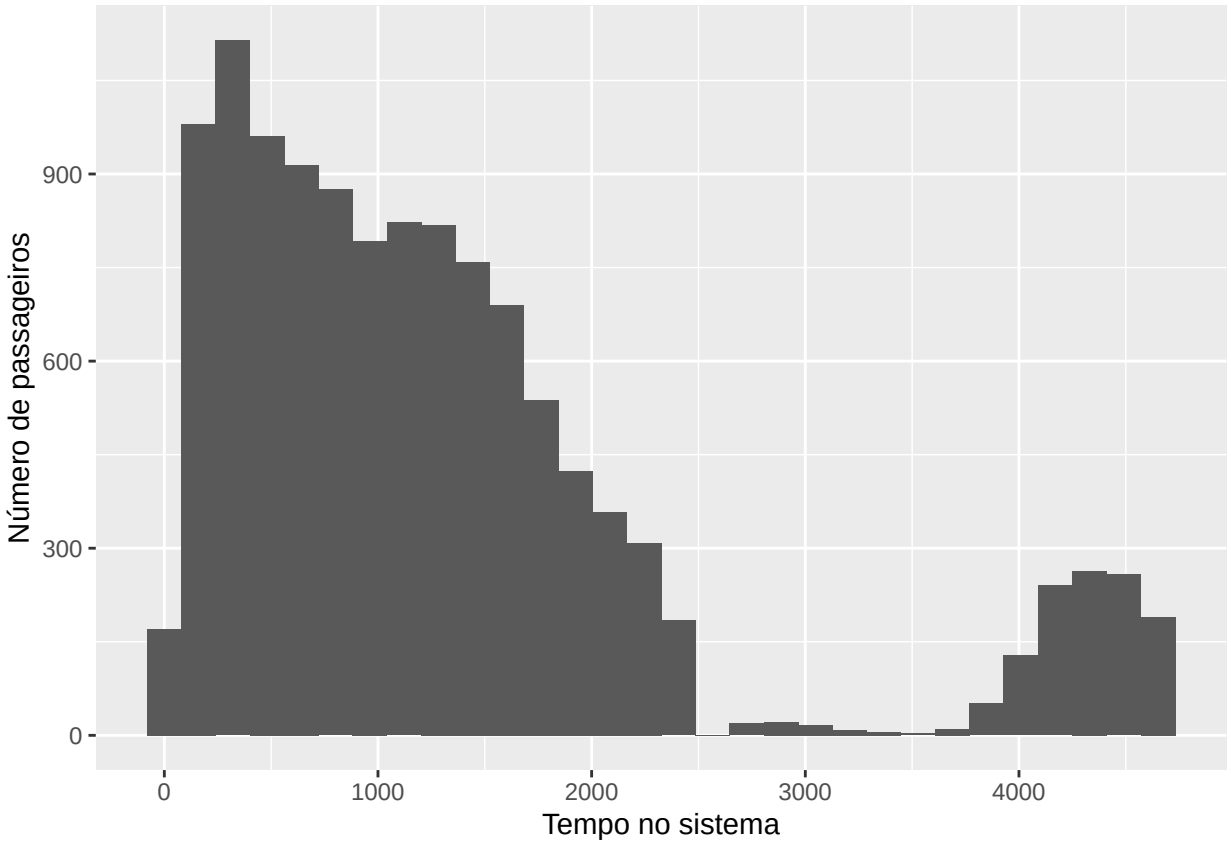



```
plot(resources, metric = "usage", item = "server", steps = TRUE)
```




```
arrivals %>%
  ggplot(aes(end_time - start_time)) +
  geom_histogram() +
  xlab("Tempo no sistema") +
  ylab("Número de passageiros")
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```




```
# número de passageiros que não conseguiram terminar o check-in
(avg_left_behind_passengers <- nrow(arrivals[!arrivals$finished,])/runs)
```

```
## [1] 0
```

```
# Cálculo do tempo de espera
arrivals <- transform(arrivals, waiting_time = end_time - start_time - activity_time)
#tempo de espera em minutos
summary(arrivals$waiting_time/60)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   0.000   7.015  16.153  20.652  26.588  76.699
```

```
#tempo de espera em minutos dos VIP
summary(arrivals[grepl("VIP Passenger", arrivals$name),]$waiting_time/60)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   0.000   4.158   9.130  10.730  16.944  32.681
```

No caso em que colocamos os passageiros *VIP* a serem atendidos nos balcões *standard*, deixando então de ter um balcão exclusivo, mas atribuindo-lhes prioridade no atendimento face aos passageiros de *Economy*, notamos que existem diferenças no "comportamento" do sistema:

- Nos gráficos de utilização dos recursos existem diversas replicações em que a percentagem de utilização da máquina de *check-in* tem um declínio até ao momento em que começam a surgir os primeiros passageiros *VIP* (aos 2400 segundos - 40 minutos), retomando então aí uma tendência crescente para o limite máximo. Já os 2 balcões de atendimento mantêm uma taxa de utilização no máximo da capacidade.

- Analisando os tempo de espera, é notório o decréscimo nas suas médias: passamos de 28.29 minutos para 20.65 minutos no caso dos passageiros *Economy*, e de 18.58 minutos para 10.73 minutos no caso dos passageiros *VIP*. Olhando para os tempos máximos de espera, nos passageiros *Economy* temos um ligeiro crescimento (de 75.37 minutos para 76.69 minutos - não relevante) e nos passageiros *VIP* sofre também um decréscimo (passa dos 38.48 minutos para os 32.68 minutos).

Nesta última simulação é fácil concluir que a criação de balcões de atendimento exclusivos para determinado segmento de clientes não é garantia de um melhor aproveitamento dos recursos e de uma melhor prestação de serviço. O que os dados nos indicam neste caso é que a construção de um sistema com prioridades diferenciadas por segmento nos traz uma melhor optimização do sistema.