

Nama: Muhammad Hanif

NIM: 202210370311265

Kelas: Pemodelan dan Simulasi Data B

Sistem Pengantaran Makanan Online

Pada implementasi model ini, sistem yang disimulasikan adalah layanan pengantaran makanan. Model terdiri dari komponen, yaitu pelanggan (customer) yang melakukan pesanan, agents dalam case ini adalah para driver yang mengantarkan pesanan, serta waktu tunggu yang merupakan durasi pelanggan menunggu driver hingga makanan diantar. Pemodelan menggunakan SimPy library python untuk melakukan simulasi berbasis event dalam pengelolaan antrian dan proses pengantaran makanan. Dalam simulasi ini, sistem dijalankan dengan berbagai skenario tingkat kedatangan pesanan dan waktu layanan guna mengukur efisiensi operasional. Proses ini mencakup pelanggan yang membuat pesanan, driver yang mengambil pesanan dari antrian, dan pengantaran makanan dengan durasi acak yang mengikuti distribusi eksponensial. Setelah simulasi dijalankan, analisis dilakukan untuk menghitung rata-rata waktu tunggu pelanggan, total waktu pengantaran pesanan, serta jumlah pesanan yang mengalami keterlambatan melebihi batas yang telah ditentukan. Dari hasil simulasi, dapat diperoleh wawasan mengenai efisiensi sistem, seperti apakah jumlah driver mencukupi untuk memenuhi permintaan pelanggan.

Source code: https://github.com/muhhanif27/PSD_TEORI

```
1 import simpy
2 import random
3 import numpy as np
4
5 # Parameter Simulasi
6 RANDOM_SEED = 42 # Seed untuk hasil random yang konsisten
7 JUMLAH_DRIVER = 5 # Jumlah driver tersedia dalam sistem
8 KECEPATAN_PESANAN_MASUK = 10 # Rata-rata jumlah pesanan masuk per jam
9 WAKTU_ANTAR = 15 # Waktu rata-rata antar pesanan dalam menit
10 DURASI_SIMULASI = 120 # Durasi total simulasi dalam menit
11 BATAS_KETERLAMBATAN = 30 # Pesanan dianggap terlambat jika lebih dari 30 menit
12
13 # Variabel untuk menyimpan data hasil simulasi
14 waktu_tunggu = [] # List untuk mencatat waktu tunggu pesanan
15 waktu_total = [] # List untuk mencatat total waktu pengantaran
16 pesanan_terlambat = 0 # Counter untuk jumlah pesanan yang mengalami keterlambatan
17
18 class GoFood:
19     """Kelas untuk merepresentasikan sistem pengantaran makanan."""
20
21     def __init__(self, env, jumlah_driver, waktu_antar):
22         self.env = env
23         self.drivers = simpy.Resource(env, jumlah_driver) # Sumber daya driver
24         self.waktu_antar = waktu_antar
25         self.antrian = simpy.Store(env) # Struktur data untuk menyimpan antrian pesanan
26
27     def antar_pesanan(self, pesanan, driver_id, waktu_masuk):
28         """Simulasi proses pengantaran makanan oleh driver tertentu."""
29
30         # Waktu pengantaran pesanan diambil secara acak dari distribusi eksponensial
31         waktu_pengantaran = random.expovariate(1 / self.waktu_antar)
32         yield self.env.timeout(waktu_pengantaran) # Driver melakukan pengantaran
33
34         # Hitung total waktu dari pemesanan hingga makanan diterima pelanggan
35         total_waktu = self.env.now - waktu_masuk
36         waktu_total.append(total_waktu)
37
38         # Jika total waktu lebih dari batas keterlambatan, hitung sebagai pesanan terlambat
39         global pesanan_terlambat
40         if total_waktu > BATAS_KETERLAMBATAN:
41             pesanan_terlambat += 1
42
43         print(f"[{self.env.now:.2f}] Driver {driver_id} mengantar {pesanan} dalam {waktu_pengantaran:.2f} menit. (Total: {total_waktu:.2f} menit)")
```

Kelas GoFood mensimulasikan layanan antar makanan dengan sejumlah driver yang tersedia. Pesanan yang masuk akan menunggu driver yang kosong, lalu diantar dengan waktu pengantaran yang bervariasi. Setelah makanan sampai, sistem mencatat total waktu sejak pemesanan hingga diterima pelanggan. Jika lebih dari 30 menit, pesanan dianggap terlambat. Hasil simulasi ini bisa membantu melihat apakah jumlah driver cukup dan seberapa sering terjadi keterlambatan.

```

1  def pelanggan(env, nama_pesanan, gofood):
2      """Proses pelanggan melakukan pemesanan."""
3
4      waktu_masuk = env.now # Catat waktu pesanan masuk ke sistem
5      print(f"[{waktu_masuk:.2f}] {nama_pesanan} masuk.")
6
7      with gofood.drivers.request() as request:
8          yield request # Menunggu driver tersedia
9
10         # Hitung waktu tunggu hingga ada driver yang mengambil pesanan
11         waktu_tunggu_pesanan = env.now - waktu_masuk
12         waktu_tunggu.append(waktu_tunggu_pesanan)
13
14         # ID driver yang menangani pesanan (berdasarkan jumlah driver yang sedang digunakan)
15         driver_id = gofood.drivers.count
16         print(f"[{env.now:.2f}] {nama_pesanan} diambil oleh Driver {driver_id} setelah menunggu {waktu_tunggu_pesanan:.2f} menit.")
17
18         # Proses pengantaran pesanan
19         yield env.process(gofood.antar_pesanan(nama_pesanan, driver_id, waktu_masuk))
20
21 def generator_pesanan(env, gofood, kecepatan_pesanan):
22     """Menghasilkan pesanan masuk secara acak berdasarkan distribusi eksponensial."""
23
24     id_pesanan = 1
25     while True:
26         # Waktu antar pesanan berikutnya dihitung secara acak berdasarkan kecepatan pesanan masuk
27         yield env.timeout(random.expovariate(1 / kecepatan_pesanan))
28
29         # Proses pesanan baru
30         env.process(pelanggan(env, f"Pesanan {id_pesanan}", gofood))
31         id_pesanan += 1

```

Fungsi pelanggan() mensimulasikan pelanggan yang melakukan pemesanan makanan. Ketika pesanan masuk, waktu dicatat dan dicetak di fungsi print(). Jika semua driver sedang sibuk, pesanan akan menunggu hingga ada driver yang tersedia. Setelah driver mengambil pesanan, waktu tunggu dihitung, lalu pesanan diteruskan ke proses pengantaran oleh driver. Sementara itu, fungsi generator_pesanan() berfungsi membuat pesanan baru secara acak berdasarkan distribusi eksponensial, yang mencerminkan waktu antar kedatangan pesanan. Setiap pesanan diproses secara berkelanjutan pada simulasi.

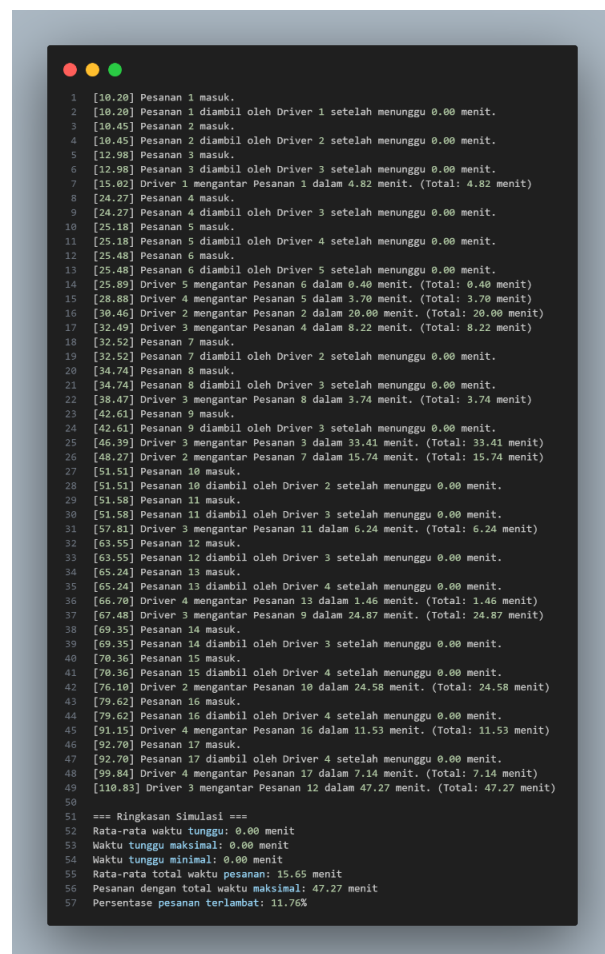
```

1  # Inisialisasi simulasi
2  random.seed(RANDOM_SEED)
3  env = simpy.Environment()
4  gofood = GoFood(env, JUMLAH_DRIVER, WAKTU_ANTAR)
5
6  # simulasi
7  env.process(generator_pesanan(env, gofood, KECEPATAN_PESANAN_MASUK))
8  env.run(until=DURASI_SIMULASI) # Simulasi berjalan hingga batas waktu yang ditentukan
9
10 # Analisis performa sistem berdasarkan data simulasi
11 print("\n=== Ringkasan Simulasi ===")
12 print(f"Rata-rata waktu tunggu: {np.mean(waktu_tunggu):.2f} menit")
13 print(f"Waktu tunggu maksimal: {np.max(waktu_tunggu):.2f} menit")
14 print(f"Waktu tunggu minimal: {np.min(waktu_tunggu):.2f} menit")
15 print(f"Rata-rata total waktu pesanan: {np.mean(waktu_total):.2f} menit")
16 print(f"Pesanan dengan total waktu maksimal: {np.max(waktu_total):.2f} menit")
17 print(f"Persentase pesanan terlambat: {(pesanan_terlambat / len(waktu_total)) * 100:.2f}%")
18

```

Bagian ini menginisialisasi simulasi dengan menetapkan seed untuk hasil acak yang konsisten, lalu membuat lingkungan simulasi menggunakan `simpy.Environment()`. Sistem GoFood juga dibuat dengan jumlah driver dan waktu antar yang telah ditentukan. Simulasi dimulai dengan menjalankan proses `generator_pesanan()`, yang terus menghasilkan pesanan baru, lalu berjalan hingga batas waktu simulasi tercapai. Setelah simulasi selesai, program menganalisis performa sistem dengan menghitung rata-rata, maksimum, dan minimum waktu tunggu serta total waktu pengantaran pesanan. Selain itu, program menghitung persentase pesanan yang mengalami keterlambatan, memberikan wawasan tentang efisiensi layanan pengantaran dalam simulasi ini.

Hasil / Output Program:



```
1 [10.20] Pesanan 1 masuk.
2 [10.20] Pesanan 1 diambil oleh Driver 1 setelah menunggu 0.00 menit.
3 [10.45] Pesanan 2 masuk.
4 [10.45] Pesanan 2 diambil oleh Driver 2 setelah menunggu 0.00 menit.
5 [12.98] Pesanan 3 masuk.
6 [12.98] Pesanan 3 diambil oleh Driver 3 setelah menunggu 0.00 menit.
7 [15.02] Driver 1 mengantar Pesanan 1 dalam 4.82 menit. (Total: 4.82 menit)
8 [24.27] Pesanan 4 masuk.
9 [24.27] Pesanan 4 diambil oleh Driver 3 setelah menunggu 0.00 menit.
10 [25.18] Pesanan 5 masuk.
11 [25.18] Pesanan 5 diambil oleh Driver 4 setelah menunggu 0.00 menit.
12 [25.48] Pesanan 6 masuk.
13 [25.48] Pesanan 6 diambil oleh Driver 5 setelah menunggu 0.00 menit.
14 [25.89] Driver 5 mengantar Pesanan 6 dalam 0.40 menit. (Total: 0.40 menit)
15 [28.88] Driver 4 mengantar Pesanan 5 dalam 3.70 menit. (Total: 3.70 menit)
16 [30.46] Driver 2 mengantar Pesanan 2 dalam 20.00 menit. (Total: 20.00 menit)
17 [32.49] Driver 3 mengantar Pesanan 4 dalam 8.22 menit. (Total: 8.22 menit)
18 [32.52] Pesanan 7 masuk.
19 [32.52] Pesanan 7 diambil oleh Driver 2 setelah menunggu 0.00 menit.
20 [34.74] Pesanan 8 masuk.
21 [34.74] Pesanan 8 diambil oleh Driver 3 setelah menunggu 0.00 menit.
22 [38.47] Driver 3 mengantar Pesanan 8 dalam 3.74 menit. (Total: 3.74 menit)
23 [42.61] Pesanan 9 masuk.
24 [42.61] Pesanan 9 diambil oleh Driver 3 setelah menunggu 0.00 menit.
25 [46.39] Driver 3 mengantar Pesanan 3 dalam 33.41 menit. (Total: 33.41 menit)
26 [48.27] Driver 2 mengantar Pesanan 7 dalam 15.74 menit. (Total: 15.74 menit)
27 [51.51] Pesanan 10 masuk.
28 [51.51] Pesanan 10 diambil oleh Driver 2 setelah menunggu 0.00 menit.
29 [51.58] Pesanan 11 masuk.
30 [51.58] Pesanan 11 diambil oleh Driver 3 setelah menunggu 0.00 menit.
31 [57.81] Driver 3 mengantar Pesanan 11 dalam 6.24 menit. (Total: 6.24 menit)
32 [63.55] Pesanan 12 masuk.
33 [63.55] Pesanan 12 diambil oleh Driver 3 setelah menunggu 0.00 menit.
34 [65.24] Pesanan 13 masuk.
35 [65.24] Pesanan 13 diambil oleh Driver 4 setelah menunggu 0.00 menit.
36 [66.70] Driver 4 mengantar Pesanan 13 dalam 1.46 menit. (Total: 1.46 menit)
37 [67.48] Driver 3 mengantar Pesanan 9 dalam 24.87 menit. (Total: 24.87 menit)
38 [69.35] Pesanan 14 masuk.
39 [69.35] Pesanan 14 diambil oleh Driver 3 setelah menunggu 0.00 menit.
40 [70.36] Pesanan 15 masuk.
41 [70.36] Pesanan 15 diambil oleh Driver 4 setelah menunggu 0.00 menit.
42 [76.10] Driver 2 mengantar Pesanan 10 dalam 24.58 menit. (Total: 24.58 menit)
43 [79.62] Pesanan 16 masuk.
44 [79.62] Pesanan 16 diambil oleh Driver 4 setelah menunggu 0.00 menit.
45 [91.15] Driver 4 mengantar Pesanan 16 dalam 11.53 menit. (Total: 11.53 menit)
46 [92.70] Pesanan 17 masuk.
47 [92.70] Pesanan 17 diambil oleh Driver 4 setelah menunggu 0.00 menit.
48 [99.84] Driver 4 mengantar Pesanan 17 dalam 7.14 menit. (Total: 7.14 menit)
49 [110.83] Driver 3 mengantar Pesanan 12 dalam 47.27 menit. (Total: 47.27 menit)
50
51 === Ringkasan Simulasi ===
52 Rata-rata waktu tunggu: 0.00 menit
53 Waktu tunggu maksimal: 0.00 menit
54 Waktu tunggu minimal: 0.00 menit
55 Rata-rata total waktu pesanan: 15.65 menit
56 Pesanan dengan total waktu maksimal: 47.27 menit
57 Persentase pesanan terlambat: 11.76%
```

Hasil simulasi menunjukkan bahwa semua pesanan langsung diambil oleh driver tanpa waktu tunggu, menandakan jumlah driver yang tersedia cukup untuk menangani pesanan yang masuk. Rata-rata total waktu pengantaran adalah 15.65 menit, tetapi ada pesanan dengan waktu pengantaran maksimal 47.27 menit, yang melebihi batas keterlambatan 30 menit. Sebanyak 11.76% pesanan mengalami keterlambatan, yang menunjukkan adanya kendala dalam proses pengantaran meskipun ketersediaan driver memadai. Dapat disimpulkan bahwa meskipun jumlah driver mencukupi untuk menangani pesanan tanpa waktu tunggu, faktor lain seperti jarak tempuh dan waktu pengantaran yang bervariasi dapat menyebabkan keterlambatan pada beberapa pesanan. Untuk mengurangi persentase pesanan terlambat, sistem dapat mempertimbangkan optimasi rute pengantaran, penambahan driver di jam sibuk, atau pemberian estimasi waktu lebih akurat terhadap sistem.