QUEUING THEORY FINAL PROJECT

312551148 吳明憲

ITEM 1 – SCENARIO-1

- Scenario1 (M/M/1), $\rho = \lambda / \mu$:

For Waiting Time CDF, I use the formula below (wt = Waiting Time)

$W(wt) = (1 - \rho * \exp^{\wedge}(-\mu * (1 - \rho) * wt))$

```
def get_WaitingTime_theo(self):
    lo = lamda / Mu
    self.WaitingTime_theo = np.unique(sorted(self.WaitingTime))
    for i, wt in enumerate(self.WaitingTime_theo):
        self.WaitingTime_theo[i]  = (1 - lo * np.exp(-Mu * (1-lo) * wt)) * 100
```

For System Time CDF, I use the formula below (st = System Time)

$S(st) = (1 – \exp^{\wedge}(- \mu * (1 - \rho) * st))$

```
def get_SystemTime_theo(self):
    lo = lamda / Mu
    self.SystemTime_theo = np.unique(sorted(self.SystemTime))
    for i, st in enumerate(self.SystemTime_theo):
        self.SystemTime_theo[i] = (1 - np.exp(-Mu * (1 - lo) * st)) * 100
```

ITEM 1 – SCENARIO-2

- Scenario2 (M/H_2/1) Use the example of case2 (p=[0.25,0.75],):

For Waiting Time CDF, I use the formula below (y = Waiting Time)

1. Get the Laplace Transform B*(s)

$$B^*(s) = \left(\frac{1}{4}\right)\frac{\lambda}{s+\lambda} + \left(\frac{3}{4}\right)\frac{2\lambda}{s+2\lambda}$$

2. Insert the result from Step.1 to the third P-K transform equation

And I eliminate the s in the denominator, which is equivalent to the integration, which can ignore one step to do integration

$$W^*(s) = \frac{s(1-\rho)}{s-\lambda+\lambda B^*(s)}$$

ITEM 1 – SCENARIO-2

- Scenario2 (M/H_2/1) Use the example of case2 (p=[0.25,0.75],):

For Waiting Time CDF, I use the formula below (y = Waiting Time)

3.Inverse the Laplace Transform W*(y) from Step.2

w(y) = Inverse Laplace Transform(W*(y))

4.Do the integration to w(y), and we can get the W(y) : Waiting Time CDF

$$W(y) = \int_0^y w(y)\, dy$$

```python
def get_WaitingTime_theo(self):
    lo = Prob[0]*(lamda / Mu[0]) + Prob[1]*(lamda / Mu[1])
    s = Symbol('s')
    t = Symbol('t')
    B_Laplace_s = Prob[0] * (Mu[0] / (s + Mu[0])) + Prob[1] * (Mu[1] / (s + Mu[1]))
    # W_Laplace_s / s 相當於積分,因此後面不需要積分 (By Dr. Stone)
    W_Laplace_s = (1 - lo) / (s - lamda + lamda * B_Laplace_s)
    w_y = inverse_laplace_transform(W_Laplace_s, s, t)
    self.WaitingTime_theo = np.unique(sorted(self.WaitingTime))
    for i, wt in enumerate(self.WaitingTime_theo):
        self.WaitingTime_theo[i]  = w_y.subs(t, wt) * 100
```

ITEM 1 – SCENARIO-2

- Scenario2 (M/H_2/1) Use the example of case2 (p=[0.25,0.75],):

For System Time CDF, I use the formula below (y = System Time)

1. Get the Laplace Transform B*(s)

$$B^*(s) = \left(\frac{1}{4}\right)\frac{\lambda}{s+\lambda} + \left(\frac{3}{4}\right)\frac{2\lambda}{s+2\lambda}$$

2. Insert the result from Step.1 to the second P-K transform equation

$$S^*(s) = B^*(s)\boxed{\frac{s(1-\rho)}{s-\lambda+\lambda B^*(s)}} \rightarrow$$

Equal to Waiting Time Laplace Tranform W*(s)

3. Inverse the Laplace Transform S*(s) from Step.2

s(y) = Inverse Laplace Transform (S*(s))

ITEM 1 – SCENARIO-2

- Scenario2 (M/H_2/1) Use the example of case2 (p=[0.25,0.75],):

For System Time CDF, I use the formula below (y = System Time)

4. Do the Integration on s(y), and we can get the S(y) : System Time CDF

$$S(y) = \int_0^y s(y)\,dy$$

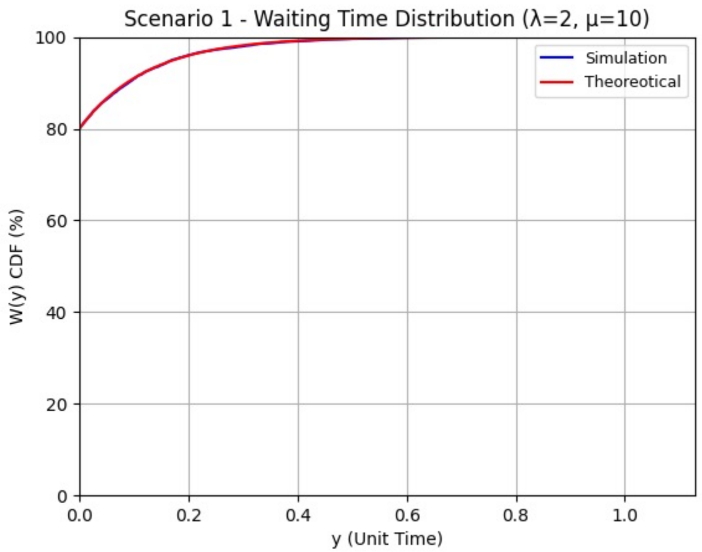
```
def get_SystemTime_theo(self):
    lo = Prob[0]*(lamda / Mu[0]) + Prob[1]*(lamda / Mu[1])
    s = Symbol('s')
    t = Symbol('t')
    B_Laplace_s = Prob[0] * (Mu[0] / (s + Mu[0])) + Prob[1] * (Mu[1] / (s + Mu[1]))
    W_Laplace_s = s * (1 - lo) / (s - lamda + lamda * B_Laplace_s)
    S_Laplace_s = B_Laplace_s * W_Laplace_s
    S_y = inverse_laplace_transform(S_Laplace_s, s, t)
    S_integrate = integrate(S_y)
    self.SystemTime_theo = np.unique(sorted(self.SystemTime))
    for i, st in enumerate(self.SystemTime_theo):
        self.SystemTime_theo[i] = S_integrate.subs(t, st) * 100
```

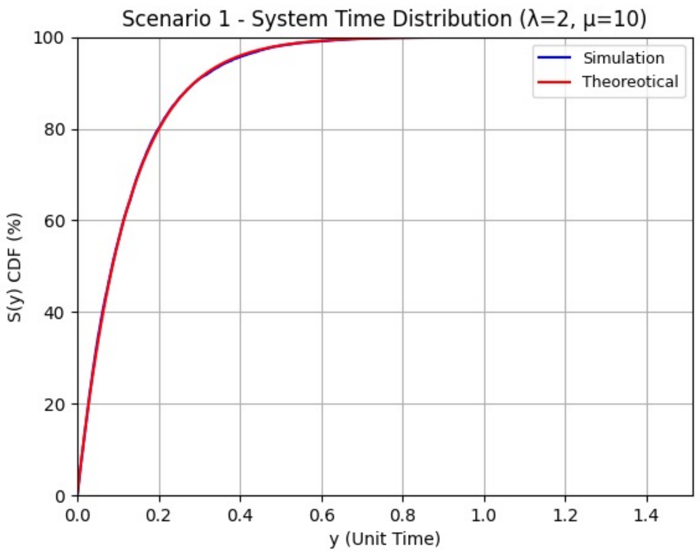
ITEM 2 – SCENARIO-1

- Case1: λ = 2, μ = 10

WaitingTime MSE: 0.003843640256692824

SystemTime MSE: 0.16352040386512556



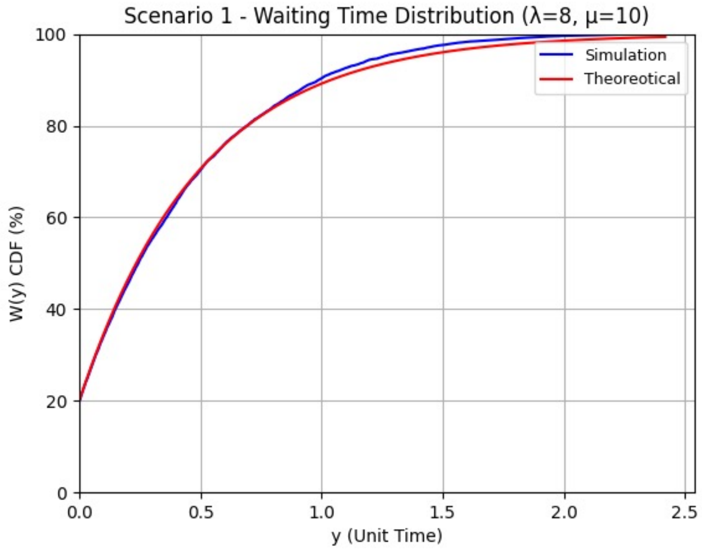


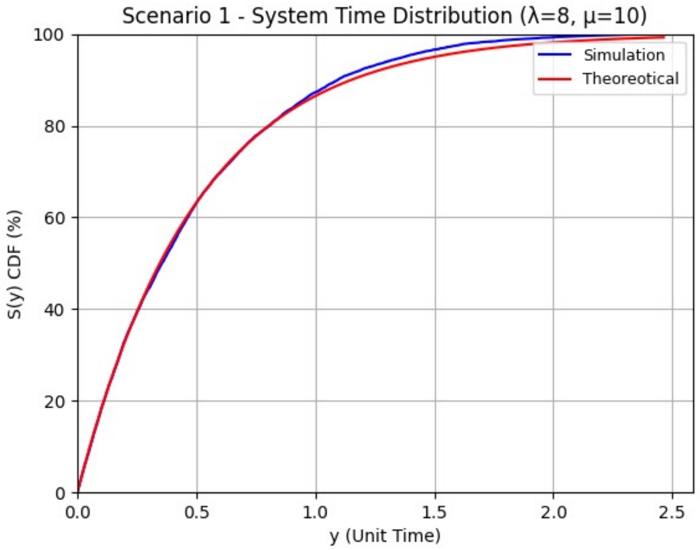
ITEM 2 – SCENARIO-1

- Case2: λ = 8, μ = 10

WaitingTime MSE: 0.46581571736205524

SystemTime MSE: 0.37834857587793536



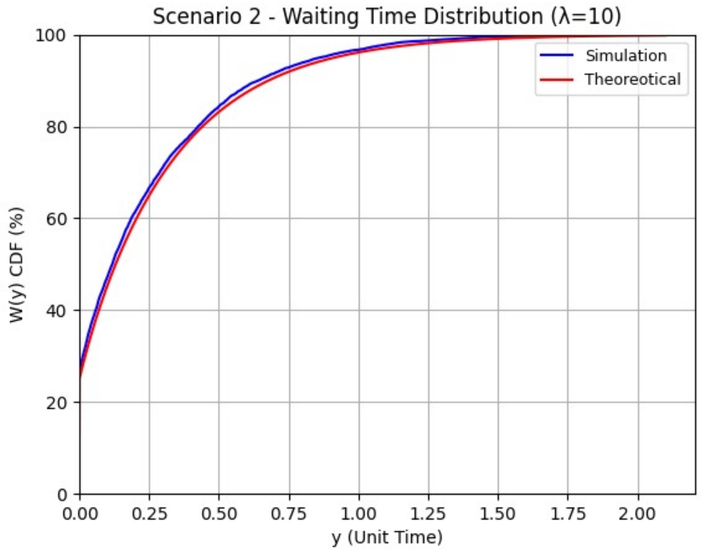


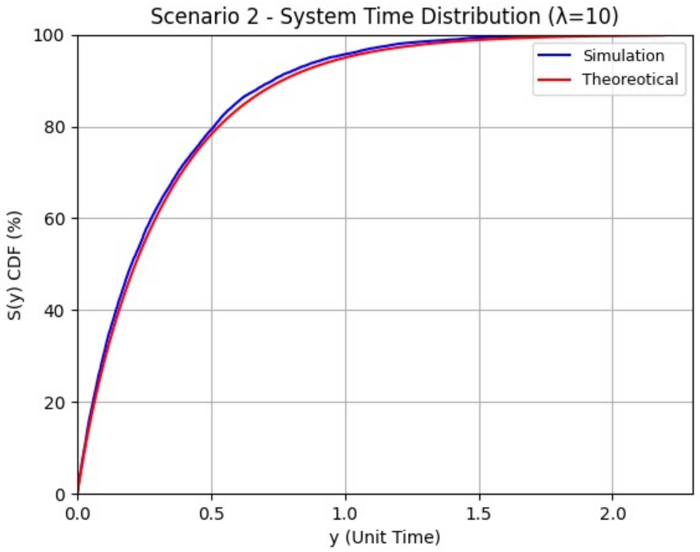
ITEM 2 – SCENARIO-2

- Case1: 50% of customers request service A, whereas 50% of customers do service B

WaitingTime MSE: 2.1207018924719256

SystemTime MSE: 2.3400690201440555



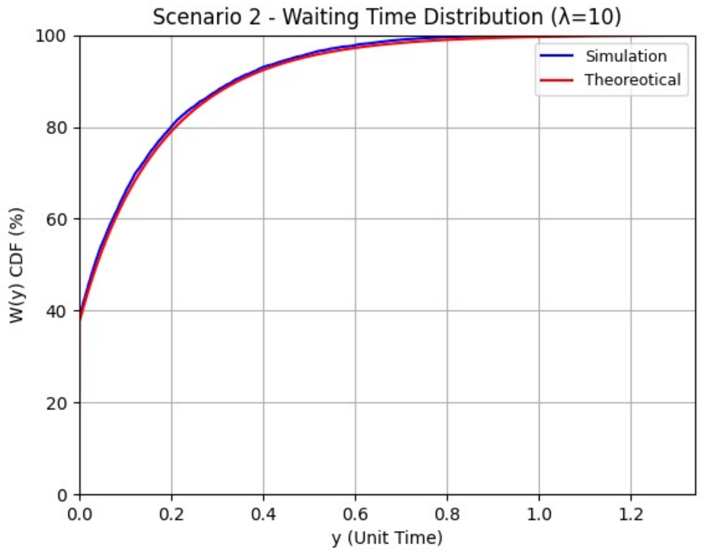


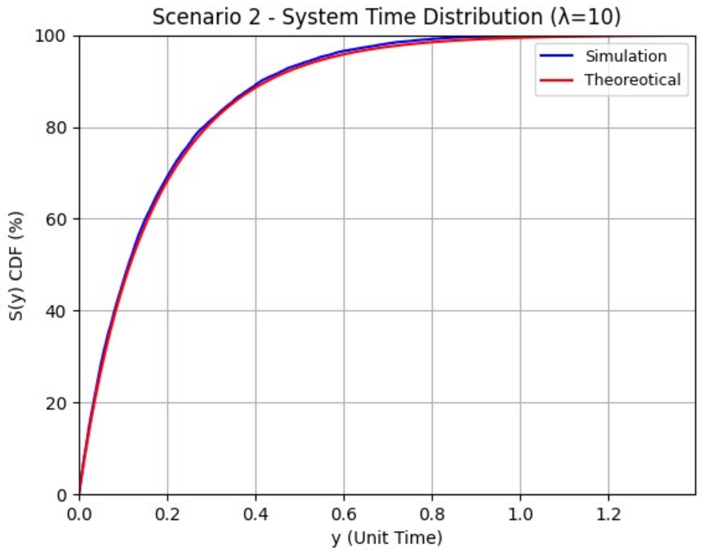
ITEM 2 – SCENARIO-2

- Case2: 25% of customers request service A, whereas 75% of customers do service B

WaitingTime MSE: 0.7451863790399651

SystemTime MSE: 0.8433457753978697





ITEM 3 – SCENARIO-1

- Case1: $\lambda = 2$, $\mu = 10$, $\rho=1/5$

WaitingTime MSE: 0.003843640256692824

SystemTime MSE: 0.16352040386512556

- Case2: $\lambda = 8$, $\mu = 10$, $\rho=4/5$

WaitingTime MSE: 0.46581571736205524

SystemTime MSE: 0.37834857587793536

In this case, we can observe that as the ρ increases, the overall waiting time also increases, and the Mean Squared Error (MSE) tends to be relatively higher.

ITEM 3 – SCENARIO-2

- Case1: 0.5 customers service A ($\mu = \lambda$), 0.5 customers service B ($\mu = 2\lambda$)

WaitingTime MSE: 2.1207018924719256

SystemTime MSE: 2.3400690201440555

- Case2: 0.25 customers service A ($\mu = \lambda$), 0.75 customers service B ($\mu = 2\lambda$)

WaitingTime MSE: 0.7451863790399651

SystemTime MSE: 0.8433457753978697

In this case, we can observe that if the service rate provided to the service with a higher probability of service is larger, the overall waiting time tends to be smaller, and the Mean Squared Error (MSE) relative to the theoretical value is also comparatively smaller.