Pontificia Universidad Católica de Chile
Escuela de Ingeniería
Departamento de Ingeniería Industrial y de Sistemas
ICS2123 – Modelos Estocásticos
Profesores Maximiliano González, Nicolás Vargas y Andrés Navarro
Ayudante Jefe Francisco Tamarín
Segundo Semestre 2024

Tarea 3:
CMTD y Proceso de Poisson

Integrantes

◇ **Integrante 1:** Vicente Lavagnino (Sección 2)

◇ **Integrante 2:** Jacinta Ortiz (Sección 4)

Instrucciones

◇ Esta tarea debe ser realizada **individualmente** o **en parejas** (pueden ser de diferentes secciones dentro de las que son coordinadas). Deben indicar en la primera plana del PDF los nombres de los/as integrantes, junto con las secciones respectivas de c/u.

◇ Tienen plazo hasta el **Martes 29 de octubre a las 23:59 hrs** para entregar sus soluciones.

◇ Deben entregar un archivo reporte, **en formato PDF**. En el archivo PDF deben entregar sus respuestas de los 3 problemas. Los códigos implementados deben adjuntarse como un archivo .zip aparte en el buzón. En el PDF deberá estar el desarrollo, resumen de resultados y correspondiente análisis. Se recomienda fuertemente utilizar LaTeX. Sin embargo, estas también pueden ser escritas en algún otro editor de texto o a mano y escaneadas, pero **debe estar ordenado y legible**, de lo contrario, se descontarán 3 décimas a la nota final de la tarea. Cualquier código que utilicen de internet debe ser debidamente citado insertando el *link* asociado.

◇ Si realizan la tarea completa en LaTeX, incluyendo todas las fórmulas y ecuaciones, tendrán **3 décimas extras**.

◇ La tarea tiene un total de 60 puntos. La nota final será calculada como la suma de las tres preguntas mediante la fórmula

$$Nota = \frac{P_1 + P_2 + P_3}{60} \cdot 6 + 1$$

Donde P_i corresponde al puntaje de la pregunta i. Se aproxima a dos decimales.

◇ Las preguntas se harán a través del foro de discusión asociado a la Tarea 3 en Canvas, por lo que les pedimos que **NO** manden dudas por mail sobre la tarea. De esta manera, todos tienen acceso a las respuestas.

◇ Seremos estrictos en que **no se aceptarán envíos por mail**. Evite problemas de internet, del sistema Canvas, de la calidad de la foto o cualquier otro que pudiese ocurrir al entregar sobre la hora. El/La alumno/a que envíe su tarea después de la hora límite, tendrá **nota 1.0**.

Problema 1 (20 puntos)

ejemplo ajedrez: tarea 2 2018-01 P1) y Tarea 2 2019-02 P1)

a)

A partir del enunciado, se definió un tablero de 8x8 , se enumero cada posición en el tablero (del 1 al 64) en conjunto a los movimientos posibles de Will y su matriz de probabilidades de transición mediante código en Pyhton (se utilizó como referencia el código de la Ayudantía 8 del Problema 3):

```python
import numpy as np
from random import choice, seed
from matplotlib import pyplot as plt
#PROBLEMA 1  (Se utilizó como referencia el código de la Ayudantía 8 P3)
#Creamos el tablero
def tablero_ajedrez():
    lista = []
    for _ in range(8):
        lista.append([0] * 8)
    return lista
# creamos un alista de las posiciones del tablero
def posiciones_validas():
    posiciones = list()
    for fila in range(8):
        for columna in range(8):
            posiciones.append([fila, columna])
    return posiciones
#definimos los movimientos posibles deacuerdo al enuncioado
def combinaciones_posicion(x, y, pos_validas):
    lista_combinaciones = [[x + 2, y ], [x + 1, y ], [x - 2, y ], [x - 1, y],  #movs horizontales
                           [x, y + 2], [x, y + 1], [x, y - 2], [x, y - 1],     #movs verticales
                           [x , y ]] #se mantiene en la posicion
    lista_valida = [x for x in lista_combinaciones if x in pos_validas]
    return lista_valida
#definimos la matriz de probabilidades
def matriz_probabilidades_transicion():
    pos_validas = posiciones_validas()

    P = np.zeros((64, 64))
    # Se recorre cada casilla del tablero
    for fila in range(8):
        for columna in range(8):
            movimientos_validos = combinaciones_posicion(fila, columna, pos_validas)
            # Calcular la probabilidad de transición para cada movimiento válido
            for movimiento in movimientos_validos:
                indice = (movimiento[0] * 8 + movimiento[1])
                P[fila * 8 + columna][indice] = 1 / len(movimientos_validos)
    # La matriz P muestra la probabilidad de que el will se mueva desde la casilla indicada
    # por la fila i a indicada por la columna j
    return P
P = matriz_probabilidades_transicion()  # Generar la matriz de probabilidades de transición
```

Debido a que la matriz resultante es de 64x64 es que analizaremos los valores de la matriz para casos específicos, es decir vectores correspondientes al estado de una casilla y sus posibles movimientos a otras casillas (mostraremos los valores donde la probabilidad de transcisión es distinta de 0):

- Vector de la casilla 1 ($Tablero_{1,1}$)

$$V_1 = \begin{bmatrix} P_{1,0} = 0.2000 & P_{1,1} = 0.2000 & P_{1,2} = 0.2000 & P_{1,8} = 0.2000 & P_{1,16} = 0.2000 \end{bmatrix}$$

  Este vector demuestra todos los posibles movimientos que puede hacer Will desde la posición incial mostrada en el ejemplo. Como se encuentra en el borde, Will sólo puede realizar 5 movimientos cada uno con probabilidad 0.2 a las siguientes casillas: (0, 1, 2, 8, 16). Y al resto de las casillas tiene probabilidad 0.

- Vector de la casilla 9 ($Tablero_{2,1}$):

$$V_9 = \begin{bmatrix} P_{9,1} = 0.1667 & P_{9,9} = 0.1667 & P_{9,10} = 0.1667 & P_{9,11} = 0.1667 & P_{9,17} = 0.1667 & P_{9,25} = 0.1667 \end{bmatrix}$$

  En este caso, como seguimos en el borde Will sólo puede realizar un total de 6 movimientos. Puede quedarse en la casilla 9 o ir a la (1, 10, 11, 17 0 25), cada movimiento con probabilidad 0.1667. El resto de las casillas tiene probabilidad 0.

- Vector de la casilla 19 ($Tablero_{3,3}$):

$$V_{19} = \begin{bmatrix} P_{19,3} = 0.1111 & P_{19,11} = 0.1111 & P_{19,17} = 0.1111 & P_{19,18} = 0.1111 & P_{19,19} = 0.1111 & P_{19,20} = 0.1111 & P_{19,21} = 0.1111 & P_{19,27} = 0.1111 & P_{19,35} = 0.1111 \end{bmatrix}$$

Este es un caso central, quiere decir que Will puede realizar 9 movimientos cada uno con probabilidad 0.1111. En este caso, como parte de la casilla 19, puede quedarse ahí o moverse a las siguientes casillas: (3, 11, 17, 18, 20, 21, 27, 35). El resto de las casillas tiene probabilidad 0.

b)

Para calcular la probabilidad de que dentro de 15 movimientos Will se encuentre frente a Chris, se utilizó el siguiente código de Pyhton:

```
# B)
#buscamos la matriz deprobabilidades de transicion de Chris
def combinaciones_posicion_cris(x, y, pos_validas):
    lista_combinaciones = [[x - 2, y - 1],
                           [x - 2, y + 1],
                           [x - 1, y - 2],
                           [x - 1, y + 2],
                           [x + 1, y - 2],
                           [x + 1, y + 2],
                           [x + 2, y - 1],
                           [x + 2, y + 1]] #se mantiene en la posicion
    lista_valida = [x for x in lista_combinaciones if x in pos_validas]
    return lista_valida

def matriz_probabilidades_transicion_cris():
    pos_validas = posiciones_validas()
    C = np.zeros((64, 64))
    # Se recorre cada casilla del tablero
    for fila in range(8):
        for columna in range(8):
            movimientos_validos = combinaciones_posicion_cris(fila, columna, pos_validas)
            # Calcular la probabilidad de transición para cada movimiento válido
            for movimiento in movimientos_validos:
                indice = (movimiento[0] * 8 + movimiento[1])
                C[fila * 8 + columna][indice] = 1 / len(movimientos_validos)
    # La matriz P muestra la probabilidad de que el will se mueva desde la casilla indicada
    # por la fila i a indicada por la columna j
    return C
cris=matriz_probabilidades_transicion_cris()
np.savetxt("matriz_probabilidades_cris.txt", cris, fmt="%.4f")

x_w, y_w = 0, 0  # Posición inicial de Will
x_c, y_c = 7, 7  # Posición inicial de Chris
inicio_will = x_w * 8 + y_w
inicio_chris = x_c * 8 + y_c

probabilidad_total = 0.0
# Iterar sobre el número de movimientos de 1 a 15
for k in range(1, 16):
    # calculamos las matrices elevadas
    P_will_k = np.linalg.matrix_power(will, k)
    P_chris_k = np.linalg.matrix_power(cris, k)
    # Iterar sobre todas las posiciones de Chris
    for x in range(8):
        for y in range(8):
            pos_chris = x * 8 + y  # Índice de la posición de Chris

            # Calcular la posicion objetivo: frente a chris
            pos_will = (x - 1) * 8 + y if x > 0 else None

            if pos_will is not None:
                # Multiplicar las probabilidades
                probabilidad = P_will_k[inicio_will, pos_will] * P_chris_k[inicio_chris, pos_chris]
                probabilidad_total += probabilidad
print(probabilidad_total)
```

Mediante este código, se calculó la matriz de transcisión de Chris para luego recorrer las matrices de ambos elevadas en k ($k \in (1, 15)$) que representan las probabilidades en cada movimiento. Para cada k, se itereró sobre el tablero de ajedrez para encontrar la probabilidad de que Will estuviera frente a Chris en cada posición de Chris de acuerdo a sus posiciones iniciales, en cada paso estas probabilidades se acumalaban en "probabilidad_total". La cual resultó 0.1838.

c)

- CMTD de Will:
  Debido a que, cualquier estado de la cadena se puede alcanzar desde cualquier otro estado en algún número de pasos **la clase es irreducible**. Por otro lado, como en cada estado existe la posibilidad de volver a este en 1 movimiento es que la cadena tiene una única clase que es recurrente positiva aperiódica. Entonces, como no hay clases recurrentes positivas periódicas en la cadena es que **existen todos los límites de las probabilidades límite** , además como no existe más de una clase recurrente es que **existe distribución límite**. Ahora, como la CMTD es irreducible y aperiodica, entonces el proceso **tiene distribución estacionaria**.

- CMTD de Chris:
  Debido al movimiento en "L"de Chris podemos notar que desde cualquier estado puede volver a este en 2 movimientos, por lo que la cadena tiene una única clase que es recurrente positiva periódica con d=2. Esto además, nos quiere decir que todos los estados se comunican entre sí en un número de pasos por lo que **la clase es irreducible**. Por otro lado, como hay una clase recurrente positiva periódica en la cadena es que **no existen todos los límites de las probabilidades límite** y por consecuencia **no existe distribución límite**. Para definir si existe distribución estacionaria buscamos si el siguiente sistema tiene solución:

$$\pi = P^T \pi$$
$$\sum_{i \in I} \pi_i = 1$$

Con el siguiente código de Pyhton se demostró que el sistema tiene solución y por lo tanto **existe distribución estacionaria**.

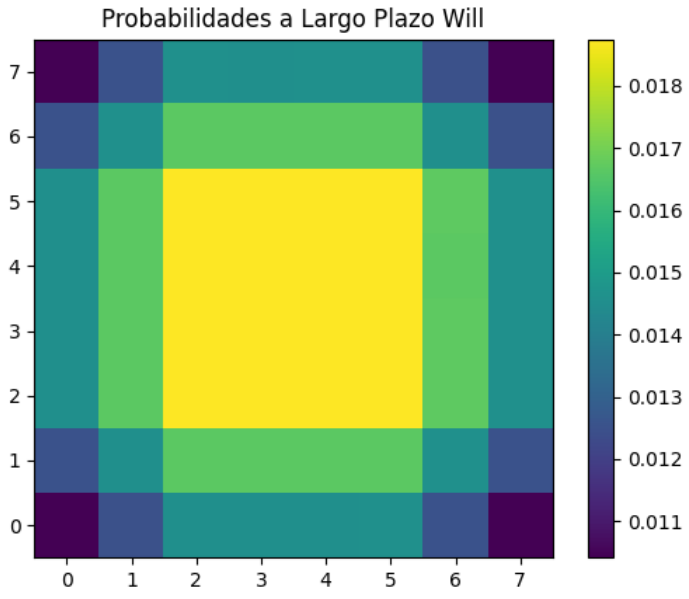
```
## C

P = matriz_probabilidades_transicion_cris()
A = np.transpose (np.identity(64) - np.matrix(P))
A = np.vstack([A,[1 for _ in range(64)]])
b = np.transpose(np.array([0 for i in range(64)] + [1]) )
pi = np.linalg.lstsq(A, b, rcond=None)[0]
pi_reshape = np.reshape(pi, (8,8))
print(pi_reshape)
```

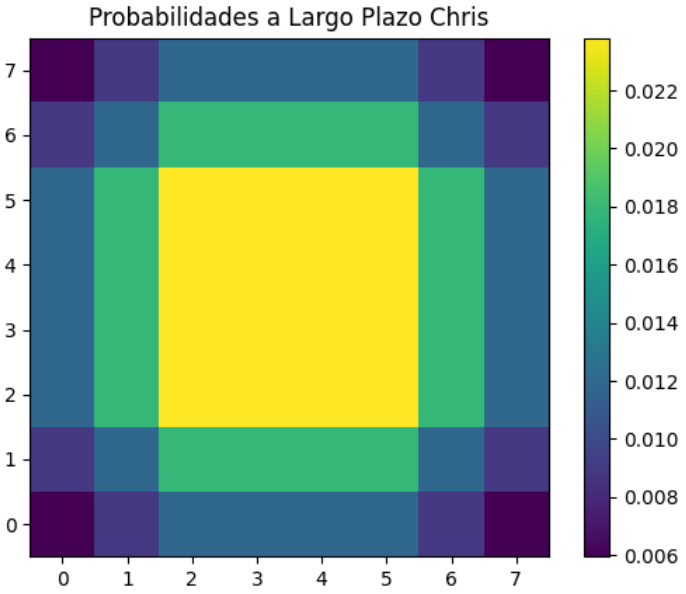
d)

◇ Mediante el sistema de la distribución estacionaria se calculó con el siguiente código de Pyhton (continuación del inciso anterior) la probabilidad a largo plazo de que Will y Chris estén en la casilla del oscar (cada uno). Se obtuvo que la probabilidad de que Will esté en la casilla del oscar es de un 1.45 % y de Chris un 1.19 %.

```
##D

W = matriz_probabilidades_transicion()
A_W = np.transpose (np.identity(64) - np.matrix(W))
A_W = np.vstack([A_W,[1 for _ in range(64)]])
b_W = np.transpose(np.array([0 for i in range(64)] + [1]) )
pi_W = np.linalg.lstsq(A_W, b_W, rcond=None)[0]
pi_reshape_W = np.reshape(pi_W, (8,8))

probabilidad_oscar_will = pi_reshape_W[6][1]
print("WILL",probabilidad_oscar_will) ##prob will

probabilidad_oscar = pi_reshape[6][1]
print("CHRIS",probabilidad_oscar) ##prob CHRIS
```

◇ Para la representación de las probabilidades de largo plazo se utilizó código de Python y se obtuvieron las siguientes gráficas:





Al ver las gráficas podemos notar que estas toman una apariencia similar, sin embargo, podemos analizar que las probabilidades de Will son más uniformes que las de Chris, ya que en el caso de Will, estas toman valores entre 0.011 a 0.018 y en cambio las de Chris toman valores desde 0.006 a 0.022. En ambos casos, es más probable encontrarlos en el centro del tablero que en las esquinas, aunque sí es más probable encontrar a Will en una esquina con 1,1 % de probabilidad que a Chris (0,6 %).
Las gráficas y los resultados coinciden con la intuición, ya que, Chris al moverse como un caballo de ajedrez, no puede llegar a todas las posiciones con la misma facilidad, especialmente en las esquinas, donde su movimiento está más restringido. Por otro lado, Will al tener más libertad para moverse a cualquier posición adyacente en el tablero, le permite tener una distribución de probabilidad más equilibrada

Problema 2 (20 puntos)

a)

Análisis Teórico:

Sea $T_{2,1}$ el tiempo de llegada del segundo bus de la compañía 1 y $T_{3,2}$ el tiempo de llegada del tercer bus de la compañía 2.

Queremos encontrar la probabilidad:

$$P(T_{2,1} < T_{3,2})$$

donde $T_{2,1} \sim \text{Gamma}(2, 10)$ y $T_{3,2} \sim \text{Gamma}(3, 20)$.

$$\begin{aligned} P(T_{2,1} < T_{3,2}) &= \int_0^\infty P(T_{2,1} < T_{3,2} = u) \cdot f_{T_{3,2}}(u)\, du \\ &= \int_0^\infty P(T_{2,1} < u) \cdot f_{T_{3,2}}(u)\, du \\ &= \int_0^\infty \left(1 - \sum_{i=0}^{1} \frac{(10u)^i e^{-10u}}{i!}\right) 400u^2 e^{-20u}\, du \\ &= 0.040740740740740744 \end{aligned}$$

Análisis Práctico:

Hacemos una simulación en python para ver como se comporata este escenario en una iteración de 10000 para evaluar si la probabilidad teórica se asemeja a la real.

```python
import numpy as np
np.random.seed(2123)

conteo_exitos = 0

for _ in range(10000):
    tiempo_1 = 0
    tiempo_2 = 0
    buses_1 = 0
    buses_2 = 0

    while buses_1 < 2 or buses_2 < 3:
        if buses_1 < 2:
            tiempo_1 += np.random.exponential(1 / 10)
            buses_1 += 1
        if buses_2 < 3:
            tiempo_2 += np.random.exponential(1 / 20)
            buses_2 += 1

    if tiempo_1 < tiempo_2:
        conteo_exitos += 1

probabilidad_empirica = conteo_exitos / 10000
print(f"Probabilidad empírica: {probabilidad_empirica}")

```

```
PROBLEMS  OUTPUT  TERMINAL  DEBUG CONSOLE  PORTS

vicentelavagnino@Vicentes-MacBook-Pro ~/L/C/O/S/2/I/T/T3> python3 p2.py
Probabilidad empírica: 0.408
```

De esta forma vemos que la probabilidad que el segundo bus de la compañía 1 llegue antes que el tercer bus de la compañía 2 es de 0.408 empíricamente.

b)

Análisis Teórico:

De 8:00 a 11:00 (3 horas), el número de llegadas en este intervalo sigue una distribución Poisson(30) donde $30 = \lambda \times 3$. 2. De 9:00 a 12:00 (3 horas), también sigue una distribución Poisson(30).

Por lo tanto:

$$P(N_{8:00 \to 11:00} = 10) = \frac{e^{-30} \cdot 30^{10}}{10!}$$

$$P(N_{9:00 \to 12:00} = 40) = \frac{e^{-30} \cdot 30^{40}}{40!}$$

Dado que los dos intervalos son independientes, la probabilidad conjunta es:

$$P(N_{8:00 \to 11:00} = 10 \text{ y } N_{9:00 \to 12:00} = 40) = P(N_{8:00 \to 11:00} = 10) \cdot P(N_{9:00 \to 12:00} = 40)$$

$$P(N_{8:00 \to 11:00} = 10 \text{ y } N_{9:00 \to 12:00} = 40) = \frac{e^{-30} \cdot 30^{10}}{10!} \cdot \frac{e^{-30} \cdot 30^{40}}{40!} = 0.0000002123.$$

Análisis Práctico:

Para este caso, tuvimos que hacer una simulación con 10000000 para obtener un valor distinto de 0. Esto se debe a que estamos trabajando con un caso extremadamemte improbabble por lo que es necesario simularlo muchas veces para obtener un caso favorable de manera aleatoria.

```python
# PARTE B
from numpy.random import poisson
np.random.seed(2123)

conteo_exitos = 0
n_simulaciones = 10000000

for _ in range(n_simulaciones):
    buses_intervalo1 = np.random.poisson(10 * 3)
    buses_intervalo2 = np.random.poisson(10 * 3)

    if buses_intervalo1 == 10 and buses_intervalo2 == 40:
        conteo_exitos += 1

probabilidad_empirica = conteo_exitos / n_simulaciones

# Resultados
print(f"Probabilidad empírica: {probabilidad_empirica}")
```

```
PROBLEMS  OUTPUT  TERMINAL  DEBUG CONSOLE  PORTS
vicentelavagnino@Vicentes-MacBook-Pro ~/L/C/O/S/2/I/T/T3> python3 p2.py
Probabilidad empírica: 4e-07
vicentelavagnino@Vicentes-MacBook-Pro ~/L/C/O/S/2/I/T/T3>
```

De esta forma vemos que la probabilidad teórica y empírica es practicamente 0.

c)

Análisis Teórico:

Sea T el tiempo de llegada del pasajero en horas desde las 19:30. El tiempo de espera W entonces:

$$W = 0.5 - T,$$

donde 0.5 horas corresponde al intervalo de 30 minutos entre las 19:30 y las 20:00.

Los pasajeros llegan de acuerdo a un proceso de Poisson con tasa ajustada $\alpha_{\text{media hora}} = 25$ pasajeros en media hora, lo que significa que los tiempos entre llegadas de pasajeros son exponenciales con un tiempo medio de $\frac{1}{25}$ horas.

Debido a que estamos en un intervalo de tiempo fijo los tiempos de llegada de los pasajeros se distribuyen de manera uniforme en este intervalo en base a la carencia de memoria.

De esta forma, el tiempo de espera de cada pasajero en el intervalo de $[0, 0.5]$ horas es de:

$$E[W] = E[0.5 - T] = 0.5 - E[T].$$

$$E[T] = \frac{0 + 0.5}{2} = 0.25 \text{ horas} = 15 \text{ minutos}.$$

$$E[W] = 0.5 - 0.25 = 0.25 \text{ horas} = 15 \text{ minutos}.$$

Por lo tanto, el tiempo promedio esperado de espera para los clientes es de 15 minutos.

Análisis Práctico:

```python
# PARTE C
n_simulaciones = 1000
esperas = []

for i in range(n_simulaciones):
    tiempo_llegada = np.random.uniform(0, 30)
    tiempo_espera = max(0, 30 - tiempo_llegada)

    esperas.append(tiempo_espera)

# Tiempo promedio de espera empírico
espera_promedio = np.mean(esperas)

# Resultados
print(f"Tiempo promedio de espera en minutos: {espera_promedio}")
```

```
PROBLEMS  OUTPUT  TERMINAL  PORTS  DEBUG CONSOLE  GITLENS

vicentelavagnino@Vicentes-MBP ~/L/C/O/S/2/I/T/T3> python3 p2.py
Tiempo promedio de espera en minutos: 14.91068410925932
```

d)

Análisis Teórico:

Consideremos los dos buses en el terminal, cada uno con una capacidad de $M = 100$ pasajeros.

Dado que el tiempo de espera del Bus 1, T_1, se distribuye uniformemente en el intervalo $[15, 30]$ y el tiempo de espera del Bus 2, T_2, se distribuye exponencialmente con parámetro $\beta = \frac{1}{5}$.

Para el Bus 1: El número de pasajeros, N_1, sigue una distribución de Poisson con media $0.3 \cdot T_1$, ya que cada pasajero elige este bus con probabilidad $p = 0.3$.

Para el Bus 2: El número de pasajeros, N_2, sigue una distribución de Poisson con media $0.7 \cdot T_2$, ya que cada pasajero elige este bus con probabilidad $1 - p = 0.7$.

De esta forma, calculamos la probabilidad de que el número de pasajeros en el Bus 1 sea más de 3 veces el número de pasajeros en el Bus 2:

$$P(N_1 > 3 \times N_2)$$

$$P(N_1 > 3 \times N_2) = \sum_{n_2=0}^{\infty} P(N_1 > 3n_2 \mid N_2 = n_2) \cdot P(N_2 = n_2)$$

$$P(N_1 > 3 \times N_2) = \sum_{n_2=0}^{\infty} \left(\int_{15}^{30} \left(\sum_{k=3n_2+1}^{\infty} \frac{(0.3 \cdot t)^k e^{-0.3 \cdot t}}{k!} \right) \frac{1}{15} \, dt \right) \cdot \left(\int_0^{\infty} \frac{(0.7 \cdot t)^{n_2} e^{-0.7 \cdot t}}{n_2!} \cdot \frac{1}{5} e^{-\frac{t}{5}} \, dt \right)$$

Al llevar esta ecuación a Symbolab, obtenemos el valor de 0.4612025795666265.

Análisis Empírico:

```python
# PARTE D
n_simulaciones = 1000
conteo_exitos = 0

for _ in range(n_simulaciones):
    tiempo_espera_bus1 = np.random.uniform(15, 30)
    tiempo_espera_bus2 = np.random.exponential(5)

    pasajeros_bus1 = np.random.poisson(0.3 * 1 * tiempo_espera_bus1)
    pasajeros_bus2 = np.random.poisson(0.7 * 1 * tiempo_espera_bus2)

    if pasajeros_bus1 > 3 * pasajeros_bus2:
        conteo_exitos += 1

probabilidad_empirica = conteo_exitos / n_simulaciones
print(f"Probabilidad empírica: {probabilidad_empirica}")
```

```
PROBLEMS  OUTPUT  TERMINAL  PORTS  DEBUG CONSOLE  GITLENS

vicentelavagnino@Vicentes-MBP ~/L/C/O/S/2/I/T/T3> python3 p2.py
Probabilidad empírica: 0.44
```

e)

Análisis Teórico:
Definimos $V(t)$ como la cantidad de vendedores ambulantes que llegan hasta t que distribuye Poisson(80t) T_1 como el tiempo de espera del chofer del bus 1 que dsitribuye Uniforme(20,30) y T_2 como el tiempo de espera del chofer del bus 2 que dsitribuye Uniforme(5,15)
Queremos calcular la probabilidad condicional de que la cantidad de vendedores que intentaron vender al bus 2 sea a lo más k, dado que la cantidad de vendedores que intentaron vender al bus 1 fue exactamente n, es decir:

$$P(V_2(t) \leq k | V_1(t) = n)$$

Lo cual lo podemos reescribir:

$$P(V_2(t) \leq k | V_1(t) = n) = \frac{P(V_2(t) \leq k, P(V_1(t) = n)}{V_1 = n} = \frac{P(V_2(t_2) \leq k | t_2 = u) f_{t_2}(u) \cdot P(V_1(t_1) = n | t_2 = s) f_{t_2}(s)}{V_1 = n}$$

$$\sum_{k}^{\infty} \frac{P(V_2(u) \leq k) f_{t_2}(u) \cdot P(V_1(s) = n) t_2 = s) f_{t_2}(s)}{V_1 = n}$$

Análisis Práctico:

En particular podemos ver como esta probabilidad es 0 debido a la magnitud de las variables asociadas y la escasa probabilidad de que coincida.

```python
# PARTE E

# Parámetros del problema
a, b = 20, 30
c, d = 5, 15
n, k = 30, 20
gamma = 80


tiempo_espera_bus1_promedio = []
tiempo_espera_bus2_promedio = []
ventas_bus1_promedio = []
ventas_bus2_promedio = []

n_simulaciones = 100000

conteo_exitos = 0

for i in range(n_simulaciones):

    tiempo_espera_bus1 = int(np.random.uniform(a, b) // 1)
    tiempo_espera_bus2 = int(np.random.uniform(c, d) // 1)
    tiempo_espera_bus1_promedio.append(tiempo_espera_bus1)
    tiempo_espera_bus2_promedio.append(tiempo_espera_bus2)


    vendedores_bus1 = np.random.poisson(gamma / tiempo_espera_bus1)
    vendedores_bus2 = np.random.poisson(gamma / tiempo_espera_bus2)


    ventas_bus1_promedio.append(vendedores_bus1)
    ventas_bus2_promedio.append(vendedores_bus2)


    if vendedores_bus1 == n and vendedores_bus2 <= k:
        conteo_exitos += 1

probabilidad_empirica = conteo_exitos / n_simulaciones
print(f"Probabilidad empírica: {probabilidad_empirica}")
print(f"Tiempo promedio de espera bus 1: {np.mean(tiempo_espera_bus1_promedio)}")
print(f"Tiempo promedio de espera bus 2: {np.mean(tiempo_espera_bus2_promedio)}")
print(f"Ventas promedio bus 1: {np.mean(ventas_bus1_promedio)}")
print(f"Ventas promedio bus 2: {np.mean(ventas_bus2_promedio)}")
```

PROBLEMS OUTPUT TERMINAL PORTS DEBUG CONSOLE GITLENS

```
vicentelavagnino@Vicentes-MBP ~/L/C/O/S/2/I/T/T3> python3 p2.py
Probabilidad empírica: 0.0
Tiempo promedio de espera bus 1: 24.49272
Tiempo promedio de espera bus 2: 9.49889
Ventas promedio bus 1: 3.31744
Ventas promedio bus 2: 9.36198
vicentelavagnino@Vicentes-MBP ~/L/C/O/S/2/I/T/T3>
```

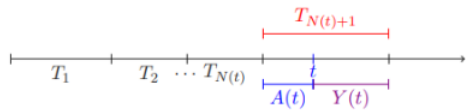
Problema 3 (20 puntos)

Parte 1

a)

Para el desarrollo del ejercicio ocuparemos la teoría de la siguiente materia de clases:

Proceso de Renovación: Paradoja de la inspección



Definamos para todo $t \geq 0$

$A(t) = t - S_N(t)$ (Exceso retroactivo o Edad)
$Y(t) = S_{N(t)+1} - t$ (Exceso futuro o Tiempo de vida residual)
$L(t) = Y(t) + A(t) = S_{N(t)+1} - S_{N(t)} = T_{N(t)+1}$ (Vida total)

En este contexto, podemos escribir al valor esperado del tiempo residual como:

$$\lim_{s\to\infty} \frac{\int_0^s A(u)\,du}{s} = \lim_{s\to\infty} \frac{\int_0^s Y(u)\,du}{s} = \frac{\mathbb{E}[T^2]}{2\mathbb{E}[T]} \tag{1}$$

Ahora, reemplazando el valor de CV_T en la expresión, buscamos demostrar lo siguiente:

$$E[T_{res}] = \frac{E[T]}{2} + \frac{\sigma_T^2}{2E[T]} \tag{2}$$

Como sabemos que $\sigma_T^2 = E[T^2] - E[T]^2$, reemplazamos en (2) y obtenemos:

$$E[T_{res}] = \frac{E[T]}{2} + \frac{E[T^2 - E[T]^2}{2E[T]} = \frac{E[T^2]}{2E[T]} \tag{3}$$

Como (1) y (3) la expresión queda demostrada.
Un análisis teórico es que en este contexto, si un pasajero llega a la parada en un momento aleatorio, es más probable que llegue durante un intervalo largo entre dos autobuses que durante un intervalo corto. Esto significa que el tiempo de espera residual promedio, $\mathbb{E}[T_{res}]$, es más alto que simplemente la mitad del tiempo promedio entre llegadas, $\frac{E[T]}{2}$, debido a la variabilidad en los tiempos de llegada.

b)

i) En el caso que exista baja variabilidad de los intervalos de llegada de buses (T), el valor esperado del tiempo residual se acerca a la mitad del valor esperado de T, de la siguiente forma:

$$E[T_{res}] \approx \frac{E[T]}{2}$$

ii) En este caso,la desviación estándar de los tiempos entre llegadas de autobuses es igual a la media de esos tiempos, es decir, la variabilidad es marginal y la expresión del valor esperado del tiempo residual resulta igual al valor esperado de los intervalos:

$$E[T_{res}] = E[T]$$

iii) En el caso que exista alta variabilidad de los intervalos de llegada de buses (T), el tiempo de espera residual promedio $E[T_{res}]$ será mucho mayor que $E[T]$, reflejando que el pasajero puede esperar mucho más tiempo cuando la variabilidad en los tiempos entre llegadas es alta. Esto ocurre porque existen intervalos de tiempo largos entre algunos autobuses, y debido a la paradoja de la inspección, es más probable que un pasajero llegue durante uno de estos intervalos largos.

$$E[T_{res}] > E[T]$$

Parte 2

c)

Primero definimos:

- $\diamond$ $N(t)$: cantidad de avistamientos contabilizados hasta t.
- $\diamond$ Y_i : cantidad de estrellas vistas en el avistamiento i
- $\diamond$ $X(t)$: cantidad total de estrellas vistas hasta t.

Según los visto en clases, $X(t)$ es un Proceso de Poisson Compuesto, por lo que E[X] se puede expresar de la siguiente forma:

$$E[X(t)] = E[Y_i] \cdot E[N(t)] \tag{4}$$

Como, sabemos que hasta las 5:30 es decir t=5.5 se han visto 3 estrellas, entonces mediante la fórmula recién expresada, buscamos la cantidad esperada del resto d ela jornada, de la siguiente forma:

$$E[X(24) - X(5.5)] = E[N(24) - N(5.5)] \cdot E[Y_1] \tag{5}$$

Como Y_i distribuye $U(1,3)$, tenemos que:

$$E[Y_1] = \frac{1+3}{2} = 4 \tag{6}$$

Ahora, como la tasa de avistamientos por hora varía segun t N(t) es un Proceso de Poisson No Homogéneo, por lo que su esperanza la calculamos integgrando bajo los valores de la tasa según t de la siguiente forma:

$$E[N(24) - N(5.5)] = \int_{5.5}^{6} \frac{10}{t+1}\, dt + \int_{6}^{18} \frac{t}{3}\, dt + \int_{18}^{24} e^{\frac{t}{6}}\, dt = 10(ln(7) - ln(6.5)) + \frac{18^2 - 6^2}{6} + 6(e^4 - e^3)$$

$$\approx 83,25$$

Reemplazamos:

$$E[X(24) - X(5.5)] = E[N(24) - N(5.5)] \cdot E[Y_1] = 83.25 \cdot 4 \approx 333$$

Finalmente, como hasta $t = 5.5$ se observaron 3 estrellas, sumamos y obtenemos el total: 333+3 = 336
La cantidad esperada de estrellas en toda la jornada es 336.

d)

Como sabemos que X es un Proceso de Poisson Compuesto con tasa variable, mediante probabilidades totales buscamos:

$$P(X(6) = 4) = \sum_{n=1}^{\infty} P(X(6) = 4 | N(6) = n) \cdot P(N(6) = n) \tag{7}$$

Si observamos los valores que puede tomar Y_i podemos notar que existen dos casos para la cantidades de avistamientos posibles para que la cantidad de estrellas sea exactamente 4. La cantidad de avistamientos puede ser 2 o 4. Entonces reemplazamos en la ecuación (4):

$$P(X(6) = 4) = \sum_{n=2}^{4} P(\sum_{i=1}^{n} Y_i = 4) \cdot P(N(6) = n) \tag{8}$$

Caso 1 Tener 2 avistamientos. En este caso puede ser que sean 2 estrellas por avistamiento o 3 y 1 estrellas. Esto quiere decir que tenemos 3^2 opciones posibles de las cuales nos sirven sólo tres (2-2,3-1, 1-3). Es decir:

$$P(Y_1 + y_2 = 4) = \frac{3}{9}$$

Caso 2 Tener 4 avistamientos de 1 estrella cada uno. Esto quiere decir que tenemos 3^4 opciones posibles de las cuales nos sirve sólo una (1-1-1-1). Es decir:

$$P(Y_1 + Y_2 + Y_3 + Y_4 = 4) = \frac{1}{81}$$

Ahora, de acuerdo a la materia, sabemos que:

$$P(N(6) = k) = \frac{m(6)^k e^{-m(6)}}{k!}$$

Definimos $m(0, 6)$:

$$m(6) = \int_0^6 \lambda(t)dt = \int_0^6 \frac{10}{t+1}dt = 10ln(6) \approx 17,91$$

Finalmente, reemplazamos en (5) con los valores obtenidos y obtenemos la probabilidad de que hasta las 6:00 se observen exactamente 4 estrellas es:

$$P(X(6) = 4) = \frac{3}{9} \cdot \frac{17,91^2 e^{-17,91}}{2!} + \frac{1}{81} \cdot \frac{17,91^4 e^{-17,91}}{4!} = 0.00028$$

e)

Del enunciado, sabemos que N(23)-N(13) es al menos 200 y buscamos:

$$P(N(16) - N(6) < 90 | N(23) - N(13) \geq 200) \tag{9}$$

Podemos escribirlo de la siguiente manera:

$$\sum_{j=0}^{89} \sum_{k=0}^{j} \frac{P(N(13) - N(6) = j - k, N(16) - N(13) = k, N(23) - N(16) \geq 200 - k}{N(23) - N(13) \geq 200}$$

Aplicando incrementos independientes:

$$\sum_{j=0}^{89} \sum_{k=0}^{j} \sum_{i=200-k}^{\infty} \frac{P(N(13) - N(6) = j - k) \cdot P(N(16) - N(13) = k) \cdot P(N(23) - N(16) = i)}{\sum_{l=400}^{\infty} N(23) - N(13) = l} \tag{10}$$

Como sabemos que

$$P(N(t) - N(s) = k) = \frac{m(s,t)^k e^{-m(s,t)}}{k!}$$

y

$$m(x, y) = \int_x^y \lambda(t)dt$$

Calculamos m para cada sección:

$$m(6, 13) = \int_6^{13} \frac{t}{3}dt = 22.17$$

$$m(13, 16) = \int_{13}^{16} \frac{t}{3}dt = 14.5$$

$$m(16, 23) = \int_{16}^{18} \frac{t}{3}dt + \int_{18}^{23} e^{\frac{t}{6}}dt = 168.12$$

$$m(13, 23) = \int_{13}^{18} \frac{t}{3}dt + \int_{18}^{23} e^{\frac{t}{6}}dt = 182.62$$

Finalmente reemplazamos en (7):

$$P(N(16) - N(6) < 90 | N(23) - N(13) \geq 200) = \sum_{j=0}^{89} \sum_{k=0}^{j} \sum_{i=200-k}^{\infty} \frac{\frac{22.17^{j-k} e^{-22.17}}{(j-k)!} \cdot \frac{14.5^k e^{-14.5}}{k!} \cdot \frac{168.12^i e^{-168.12}}{i!}}{\sum_{l=400}^{\infty} \frac{182.62^l e^{-182.62}}{l!}}$$