SEJONG UNIVERSITY
VISION 2030 WORLD TOP100 UNIVERSITY

P.E.L.
ROUGH CONSENSUS & RUNNING CODE

대기행렬의 기초
- 4장. 복수서버 대기행렬 시스템(2) -

2025.08.08.

Jihye Kim
jihye@pel.sejong.ac.kr
Protocol Engineering Lab., Sejong University

CONTENTS

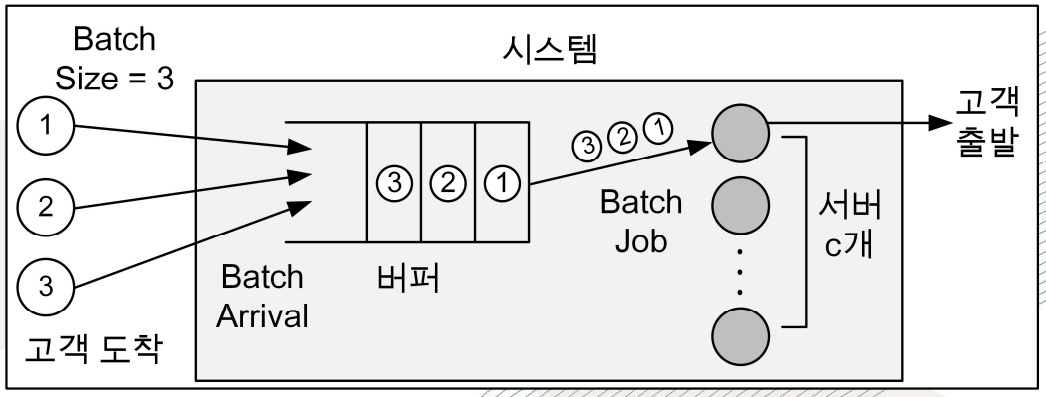
Batch Job을 가진 $M^X/M/c/c$ 큐

I

I 1 Batch Job을 가진 $M^X/M/c/c$ 큐

A. 개요

- Batch Size(일괄 크기): 한 번에 시스템에 들어오는 고객 수 혹은 한 명의 고객이 가지고 들어오는 일량의 크기
  - e.g., 한 번 도착 시 고객 3명이 동시에 들어오는 경우에 Batch Size는 3, 고객 1명이 동시에 4개의 서류를 제출하는 경우에 Batch Size는 4
- Batch Job(일괄 작업): 여러 개의 작업을 일괄적으로 모아서 동시에 처리하는 방식
  - e.g., 동시에 도착한 고객 5명을 한 번에 처리하는 것, 하루 동안 들어온 요청을 모아서 매일 자정에 한 번에 실행하는 것
- Batch Arrival(집단 도착): 한 번에 여러 명의 고객이 동시에 도착하거나, 한 명의 고객이 여러 개의 일량을 갖고 동시에 도착하는 경우
  - e.g., 한 번 도착 시 고객 3명이 동시에 들어오는 경우, 고객 1명이 동시에 4개의 서류를 제출하는 경우



(그림 1) Batch 관련 주요 용어

I 1 Batch Job을 가진 $M^X/M/c/c$ 큐

$M^X/M/c/c$

고객의 도착과정 / 고객의 서비스과정 / 서버 수 / 수용 가능한 고객 수

B. $M^X/M/c/c$ 큐의 개요

- 정의
  - X개의 일량을 가진 고객 도착과정이 포아송분포를 따르고, 고객 서비스과정이 지수분포를 따르며 서버의 수와 수용 가능한 고객 수가 동일한 복수서버 대기행렬 시스템

- 특징
  - 고객이 도착하는 순서대로(FIFO, First-In First-Out) 서비스함
  - 고객 도착은 집단 단위(Batch Arrival)로 이루어짐
  - 대기열이 존재하지 않으므로 고객의 브러킹확률(Blocking Probability)이 중요함

- 켄달(Kendall) 표기방식
  - 고객 도착과정(Arrival Process): 일량의 크기가 X일 때의 고객 도착이 포아송분포(Possion Distribution, 표기 M^X)를 따름
    - λ: 고객의 단위시간(Unit Time)당 평균도착률(Average Arrival Rate)
  - 고객 서비스과정(Service Process): 지수분포(Markov, 표기 M)를 따름
    - μ: 하나의 서버에 의한 고객당 평균서비스율(Average Service Rate) (서버 관점)
    - r: 한 명의 고객이 요구하는 평균서비스율(Average Service Rate) (고객 관점)
  - 서버의 수: 동시에 서비스가 가능한 서버가 c개
    - K: 한 명의 고객이 동시에 점유 가능한 최대 서버의 수 (단, $1 \leq K \leq c$)
  - 시스템의 수용 가능한 고객 수: 서버의 수와 동일한 c명

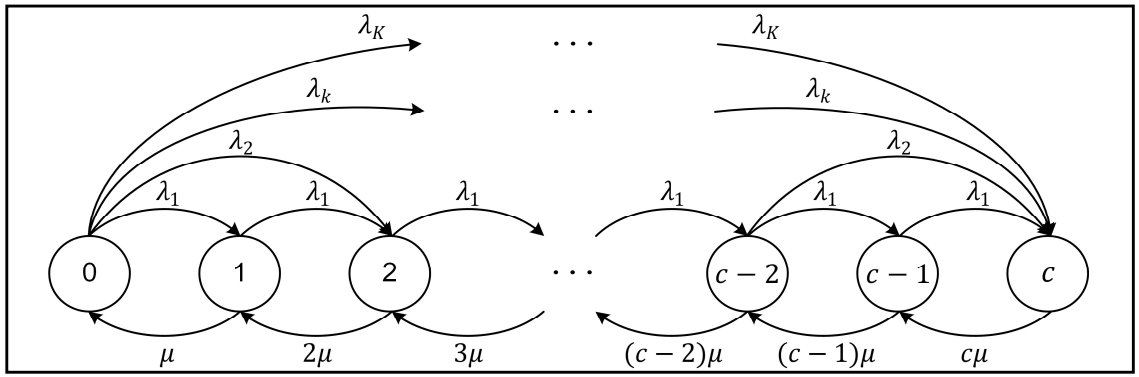
C. $M^X/M/c/c$ 큐의 상태확률

- 파라미터 정의
  - 한 명의 고객이 요구하는 평균서비스율(Average Service Rate): r
  - 시스템의 총 서비스율(Service Rate): $C = rc$
  - 일괄 크기(Batch Size)가 k일 확률: q_k (단, $\sum_{k=1}^{K} q_k = 1,\ 1 \le k \le K \le c$)
  - 일괄 크기(Batch Size)가 k인 고객의 집단 도착(Batch Arrival)에 대한 평균도착률(Average Arrival Rate): $\lambda_k = q_k\lambda$
  - 한 번의 집단 도착(Batch Arrival)에서 평균 도착 고객 수(=고객당 평균일량): $\beta = \sum_{k=1}^{K} kq_k$

특정 고객이 서비스를 받기 원하는 일(Job)이 여러 개가 동시에 도착하고,
그때의 일량이 랜덤하므로 시스템 상태의 변화를 예측하기 어려움

C. $M^X/M/c/c$ 큐의 상태확률

- 상태전이방정식(State Transition Equation)



(그림 2) $M^X/M/c/c$ 큐 상태전이도

- 상태 $\mathbf{0}$에서 옮겨갈 수 있는 상태는 상태 $\mathbf{1}$부터 상태 $\boldsymbol{K}$까지 모두 $\boldsymbol{K}$개가 있음
- $\boldsymbol{\lambda_k}$: 시스템에 고객이 일량 $\boldsymbol{k}$를 가지고 도착하여, 상태가 $\boldsymbol{k}$이 될 확률
- $\boldsymbol{k\mu}$: 시스템에 고객 $\boldsymbol{k}$명이 있다가 서비스를 받고 나가면서 상태가 $\boldsymbol{k-1}$이 될 확률

$\boldsymbol{K}$: 한 고객이 동시 요청 가능한 최대 서버 수
$\boldsymbol{p_m}$: 시스템에 고객이 $\boldsymbol{m}$명 차 있는 상태의 확률

- 경우#1: 상태 0과 상태 1 사이의 상태전이
  - $\boldsymbol{\lambda p_0 = \mu p_1}$, $(\lambda = \sum_{k=1}^{K} \lambda_k)$
- 경우#2: 상태 1 이후의 모든 상태 간 상태전이
  - $\left(\boldsymbol{m\mu} + \sum_{\boldsymbol{k=1}}^{\boldsymbol{min(c-m,K)}} \boldsymbol{\lambda_k}\right) \boldsymbol{p_m} = \sum_{\boldsymbol{k=1}}^{\boldsymbol{min(m,K)}} \boldsymbol{\lambda_k p_{m-k}} + \boldsymbol{(m+1)\mu p_{m+1}}$, $(1 \le m \le c-1)$
- 경우#3: 상태 c와 그외의 모든 다른 상태 간 상태전이
  - $\boldsymbol{c\mu p_c} = \sum_{\boldsymbol{k=1}}^{\boldsymbol{K}} \boldsymbol{\lambda_k p_{c-k}}$

C. $M^X/M/c/c$ 큐의 상태확률

- 시스템 상태확률(State Probability)
  - $\boldsymbol{p_1 = \frac{\lambda}{\mu} p_0}$
  - $\boldsymbol{p_{m+1} = \frac{\left(m\mu + \sum_{k=1}^{min(c-m,K)} \lambda_k\right) p_m - \sum_{k=1}^{min(m,K)} \lambda_k p_{m-k}}{(m+1)\mu}}$, $1 \le m \le c-1$

증명#1

상태전이방정식 경우#1 활용

$$\lambda p_0 = \mu p_1, \qquad \boldsymbol{p_1 = \frac{\lambda}{\mu} p_0}$$

현재 시스템 고객 m명, 시스템 최대 용량 c명, 동시에 최대 K명까지 도착 가능할 때, 시스템은 최대 $c-m$명까지 더 수용 가능하나 한 번에 들어오는 최대 일량이 K명이므로, 도착 가능한 고객 수는 $\boldsymbol{min(c-m, K)}$가 됨

상태전이방정식 경우#2 활용

$$\left(m\mu + \sum_{k=1}^{min(c-m,K)} \lambda_k\right) p_m = \sum_{k=1}^{min(m,K)} \lambda_k p_{m-k} + (m+1)\mu p_{m+1}$$

p_{m+1} 계산 시, 이전 상태들 $(p_m, \dots, p_{m-K})$의 값을 이용하므로, 재귀식에 해당

$$\boldsymbol{p_{m+1} = \frac{\left(m\mu + \sum_{k=1}^{min(c-m,K)} \lambda_k\right) p_m - \sum_{k=1}^{min(m,K)} \lambda_k p_{m-k}}{(m+1)\mu}}, \quad \boldsymbol{1 \le m \le c-1}$$

시스템 상태 간에 재귀식(Recursive Formula)으로 이루어진 것을 알 수 있으며, c의 값이 커질수록 재귀의 횟수도 증가하므로 계산 복잡도가 증가함

⇒ 이를 해결하기 위해 반복과정(Iterative Procedure)을 이용하여 해를 구함

C. $M^X/M/c/c$ 큐의 상태확률

- 반복과정(Iterative Procedure)
  - 어떤 문제의 해를 한 번에 계산하기 어려운 경우, 초기값을 설정하고 이전 결과를 기반으로 다음 값을 순차적으로 계산해나가는 반복적인 계산 방법

- 반복과정(Iterative Procedure) 기반 상태확률 계산
  - 가정사항: 정규화되지 않은 상태 0의 확률 값인 $\boldsymbol{p_0^* = 1}$
  - $\boldsymbol{p_1^* = \frac{\lambda}{\mu} p_0^* = \frac{\lambda}{\mu}}$,
  - $\boldsymbol{p_{m+1}^* = \frac{\left(m\mu + \sum_{k=1}^{min(c-m,K)} \lambda_k\right) p_m^* - \sum_{k=1}^{min(m,K)} \lambda_k p_{m-k}^*}{(m+1)\mu}, 1 \le m \le c-1}$
  - 확률의 정규화 조건으로부터 $\sum_{i=0}^{c} p_i = 1$이 성립함
  - $\boldsymbol{p_m = \frac{p_m^*}{\sum_{i=0}^{c} p_i^*}, 0 \le m \le c}$

***확률의 정규화 조건:**
모든 가능한 상태의 확률을 더한 값은 1이어야 함

***통신 시스템의 주요한 성능지표**

1) **고객에 대한 서비스품질 보장**: 시스템의 유한한 서버에 대해 고객의 서비스 수용 여부를 나타내는 <u>고객의 브러킹확률</u>을 일정 수준 이하로 유지하도록 설계되어야 함
(*일반적으로, 전화통신망에서 고객의 브러킹확률은 1% 이하로 유지하여야 함)
2) **시스템의 경제성**: 유한한 시스템 자원을 얼마나 효과적으로 사용하였는가를 나타내는 <u>효용(Utilization)</u>으로 나타내며, 효용이 높을수록 시스템은 경제성이 높음을 의미함

C. $M^X/M/c/c$ 큐의 상태확률

- 고객의 브러킹확률(Blocking Probability): Ω (1/2)
  - 임의의 일괄 크기(Batch Size)를 가진 고객의 시스템 도착 시, 시스템 서버 자원 제한으로 인해 서비스를 제공할 수 없게 되어 고객의 시스템 진입이 차단될 확률
  - 고객 도착 시 시스템 상태가 $c-k+1$인데, 고객이 k의 일괄 작업(Batch Job)을 요구하는 경우에 시스템 진입이 차단됨
    - e.g., 고객이 시스템 상태가 c인데 1부터 K까지의 일량을 요구하거나, 시스템 상태가 $c-1$인데 2부터 K까지의 일량을 요구하는 경우

- $\Omega = \frac{1}{\lambda}\sum_{i=0}^{K-1}\left(p_{c-i}\sum_{k=i+1}^{K}\lambda_k\right)$

$c-i$개의 서버가 점유된 상태의 확률

$i+1$ 이상의 서버를 요구하는 고객의 평균도착률

고객의 시스템 진입이 차단되는 모든 경우의 합과 고객 평균도착률의 비가 브러킹확률이 됨

r: 서버 하나가 요구하는 처리율
λ: 전체 Batch Job의 평균도착률
λ_k: 한 번에 서버 k개를 요구하는 고객의 평균도착률
K: 한 고객이 동시 요청 가능한 최대 서버 수
p_{c-i}: 시스템에 서버가 $c-i$개 점유된 상태의 확률

C. $M^X/M/c/c$ 큐의 상태확률

- 고객의 브러킹확률(Blocking Probability): Ω (2/2)

한 회사에서 비디오 서비스를 제공하며 다음과 같은 조건을 가진다. 이때 고객의 브러킹확률을 구하라.

- 총 서버 수(c): 3개
- 고객이 한 번에 요청할 수 있는 최대 서버 수(K): 2개
- 각 Batch Job의 평균도착률 (λ_k)
  - 서버 1개 요청 (λ_1): 1
  - 서버 2개 요청 (λ_2): 0.5
- 전체 Batch Job의 평균도착률 (λ): 1.5
- 시스템 상태확률
  - $p_2 = 0.2$
  - $p_3 = 0.1$

- $i = 0$인 경우
  - $c - i = 3$, 즉 $p_3 = 0.1$
  - $\sum_{k=1}^{2} \lambda_k = \lambda_1 + \lambda_2 = 1 + 0.5 = 1.5$
  - $p_3 \times \sum_{k=1}^{2} \lambda_k = 0.1 \times 1.5 = 0.15$

- $i = 1$인 경우
  - $c - i = 2$, 즉 $p_2 = 0.2$
  - $\sum_{k=2}^{2} \lambda_k = \lambda_2 = 0.5$
  - $p_2 \times \sum_{k=2}^{2} \lambda_k = 0.2 \times 0.5 = 0.1$

- $\Omega = \frac{1}{\lambda} \sum_{i=0}^{K-1} \left(p_{c-1} \sum_{k=i+1}^{K} \lambda_k \right) = \frac{0.15+0.1}{1.5} \approx 0.167$

∴ 약 16.7%의 고객이 브러킹됨

C. $M^X/M/c/c$ 큐의 상태확률

- 시스템의 효용(Utilization): Ψ (1/2)
  - 일정 관측시간 동안 고객의 시스템 사용시간(i.e., 서버 점유시간)의 비율에 해당함
    - 시스템의 서버 점유시간 $= \sum_{k=1}^{K}\left[\lambda_k \times T \times (1-\sum_{i=c-k+1}^{c} p_i) \times k \times \frac{1}{\mu}\right]$
    - $\boldsymbol{\Psi = \frac{\sum_{k=1}^{K}[k \times \lambda_k \times (1-\sum_{i=c-k+1}^{c} p_i)]}{c\mu}}$

증명#2

$$\Psi = \lim_{T\to\infty} \frac{\text{시스템의 서버 점유시간}}{T \times c} = \lim_{T\to\infty} \frac{\sum_{k=1}^{K}\left[\lambda_k \times T \times (1-\sum_{i=c-k+1}^{c} p_i) \times k \times \frac{1}{\mu}\right]}{T \times c}$$

$$= \frac{\sum_{k=1}^{K}[k \times \lambda_k \times (1-\sum_{i=c-k+1}^{c} p_i)]}{c\mu}$$

충분히 긴 시간($T \to \infty$) 동안 관측함

시스템 전체 평균서비스율에 대한 고객 총도착률의 비를 의미

고객 총도착률은 고객 유효도착률(λ_{eff})과 고객당 평균 일량(β)의 곱을 의미

$$\Psi = \frac{\sum_{k=1}^{K}[k \times \lambda_k \times (1-\sum_{i=c-k+1}^{c} p_i)]}{c\mu} = \frac{\lambda_{eff}\beta}{c\mu}$$

- 고객 유효도착률
  - $\boldsymbol{\lambda_{eff} = \frac{\sum_{k=1}^{K}[k \times \lambda_k \times (1-\sum_{i=c-k+1}^{c} p_i)]}{\beta}}$

C. $M^X/M/c/c$ 큐의 상태확률

- 시스템의 효용(Utilization): Ψ (2/2)

한 회사에서 비디오 서비스를 제공하며 다음과 같은 조건을 가진다. 이때, 시스템 효용을 구하라.

- 총 서버 수(c): 3개
- 고객이 한 번에 요청할 수 있는 최대 서버 수(K): 2개
- 각 Batch Job의 평균도착률 (λ_k)
  - 서버 1개 요청 (λ_1): 0.4
  - 서버 2개 요청 (λ_2): 0.2
- 각 서버의 서비스율 (μ): 1
- 시스템 상태확률
  - $p_2 = 0.15$
  - $p_3 = 0.1$

- 전체 batch job의 평균도착률 (λ)
  - $\lambda = \lambda_1 + \lambda_2 = 0.4 + 0.2 = 0.6$
- $k = 1$인 경우
  - $1 \times 0.4 \times (1 - 0.10) = 0.4 \times 0.9 = 0.36$
- $k = 2$인 경우
  - $2 \times 0.2 \times (1 - 0.25) = 0.4 \times 0.75 = 0.3$
- $\psi = \frac{0.36+0.3}{3\times1} = \frac{0.66}{3} = 0.22$

∴ 서버 자원의 평균 22%가 사용되고 있음

C. $M^X/M/c/c$ 큐의 상태확률

- 시스템 내 평균 고객 수(Average Number of Customers in System)
  - $L = \sum_{i=0}^{c} ip_i$ (i: 시스템 내 고객 수, p_i: 상태가 i일 확률)
    - 대기공간이 없으므로 시스템 내 고객 수와 서비스 중인 고객 수가 동일함
  - $\Psi = \frac{L}{c}$
    - 시스템이 보유한 서버 c개 중 얼마나 사용되고 있는지에 대한 비율

- 시스템 내 체재시간(Sojourn Time)
  - $M^X/M/c/c$ 큐에서는 서버 수와 큐의 수가 같으므로, 대기시간은 없고 서비스시간만 존재함
  - $S = \frac{1}{\beta\lambda_{\text{eff}}} = \frac{1}{\mu}$

고객 총도착률(μ)은 고객 유효도착률(λ_{eff})과 고객당 평균 일량(β)의 곱

한 회사에서 제공되는 전화 서비스는 다음의 조건을 가진다. 이때, 고객 1명의 평균 체재시간 S를 구하라.
- 각 서버의 평균 서비스율 (μ): 0.2
- 고객의 평균 일량 (β): 2
- 고객의 유효도착률 (λ_{eff}): 0.1

- 대기시간은 없고, 서비스시간만 존재하므로 체재시간은 다음과 같음
  - $S = \frac{1}{\beta\lambda_{\text{eff}}} = \frac{1}{2\times0.1} = \frac{1}{0.2} = 5$ 분

∴ 고객은 시스템 내에 평균 5분 동안 머무름

$M/D/N@c$ 큐

II

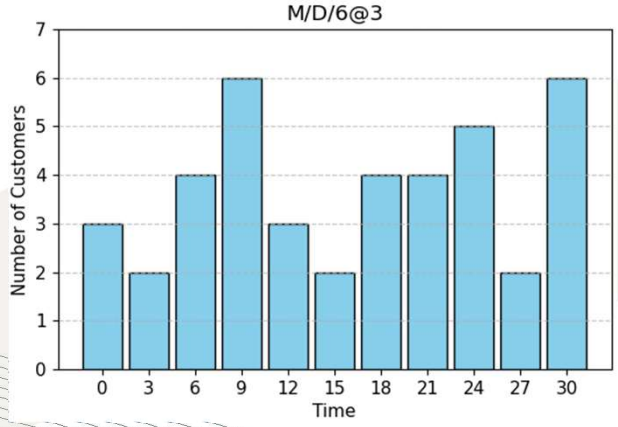
II 2 $M/D/N@c$ 큐

$M/D/N@c$
고객의 도착과정 / 고객의 서비스과정 / 수용 가능한 고객 수 / 단위 시간

A. $M/D/N@c$ 큐의 개요

- 정의
  - 고객 도착과정이 포아송분포를 따르고 고객 서비스시간이 고정된 값을 가지며, 단위시간마다 최대 N명의 고객이 동시에 서비스받는 복수서버 대기행렬 시스템
- 특징
  - 고객이 도착하는 순서대로(FIFO, First-In First-Out) 서비스함
  - 일량 보존 시스템(Work-Conserving System)이며, 주기적인 서비스를 제공함
    - 시스템 내 고객이 있는 한, 서버 자원은 쉬지 않고 지속적으로 일하는 구조
- 켄달(Kendall) 표기방식
  - 고객 도착과정(Arrival Process): 포아송분포(Poisson Distribution, 표기 M)를 따름
  - 고객 서비스과정(Service Process): 고정된(Deterministic, 표기 D) 시간을 가짐
  - 수용 가능한 고객 수: 최대 N명의 고객이 동시에 서비스를 받을 수 있으나, 단위시간 c 간격의 서비스가 주기적으로 제공됨 (표기 $N@c$)

*고객 평균도착률: λ
*고객 서비스시간: c
*시스템 안정조건: $\lambda \cdot c < N$



(그림 3) M/D/6@3 그래프

- 대기열: 손님들이 리프트 탑승 대기하는 줄
- 곤돌라: 매 3초마다 도착하여 최대 6명까지 태움
- 고정된 서비스 시간(D): 정상까지 올라가는 시간은 항상 5분
- 포아송 도착: 손님들은 무작위로 도착

- x축: 단위시간 3초에 한 번씩 30초동안 관측
- y축: 도착한 곤돌라에 탑승한 손님 수 (≤ 6)
- 매 c초(c=3)마다 오며, 도착 순서대로 최대 N명(N=6)을 태우고, 서비스 시간은 고정된 D시간(D=5분)
- (그래프 해석) 2번째 곤돌라에는 2명의 손님이 탑승함

B. $M/D/N@c$ 큐의 상태확률

- 시스템 초기 상태확률
  - 이전 사이클에서 서비스 시작 직전까지 큐 내에 대기하는 고객이 N명을 넘지 않고, 서비스 시간 동안 고객이 1명도 도착하지 않을 확률
  - $\boldsymbol{p_0 = \sum_{i=0}^{N} p_i e^{-\lambda c}}$

- 시스템 상태확률(State Probability)
  - $\boldsymbol{p_n = \sum_{i=0}^{N} \frac{p_i(\lambda c)^2}{n!} e^{-\lambda c} + \frac{p_{N+1}(\lambda c)^{n-1}}{(n-1)!} e^{-\lambda c} + \cdots + p_{N+n} e^{-\lambda c}}$

증명#3

$$p_1 = \sum_{i=0}^{N} p_i \lambda c e^{-\lambda c} + p_{N+1} e^{-\lambda c},$$

$$p_2 = \sum_{i=0}^{N} \frac{p_i(\lambda c)^2}{2} e^{-\lambda c} + p_{N+1} \lambda c e^{-\lambda c} + p_{N+2} e^{-\lambda c},$$

$$\boldsymbol{p_n = \sum_{i=0}^{N} \frac{p_i(\lambda c)^2}{n!} e^{-\lambda c} + \frac{p_{N+1}(\lambda c)^{n-1}}{(n-1)!} e^{-\lambda c} + \cdots + p_{N+n} e^{-\lambda c}}$$

- 시스템 상태가 1(p_1)인 경우
  - $\boldsymbol{p_i \lambda c e^{-\lambda c}}$: 이전 사이클에서 서비스 시작 직전까지 큐 내에 대기중인 고객이 N명을 넘지 않고, 서비스 시간 동안 1명 도착하는 경우의 확률
  - $\boldsymbol{p_{N+1} e^{-\lambda c}}$: 이전 사이클에서 서비스 시작 직전까지 큐 내에 대기중인 고객이 $N+1$명이고, 서비스 시간 동안 1명도 도착하지 않을 경우의 확률

B. $M/D/N@c$ 큐의 상태확률

- p_n의 확률생성함수(PGF, Probability Generating Function)(1/2)
  - $\boldsymbol{P(z) = \sum_{n=0}^{\infty} p_n z^n = \frac{(\sum_{i=0}^{N} p_t) z^N - \sum_{n=0}^{N} p_n z^n}{z^N e^{\lambda c(1-z)} - 1}}$

증명#4

상태확률(p_n)의 각 식에 $z^0, z^1, \ldots, z^n$을 곱함

$$P(z) = \sum_{n=0}^{\infty} p_n z^n$$
$$= \sum_{i=0}^{N} p_i e^{-\lambda c} \sum_{n=0}^{\infty} \frac{(\lambda c z)^n}{n!} + e^{-\lambda c} p_{N+1} z \sum_{n=0}^{\infty} \frac{(\lambda c z)^n}{n!} + e^{-\lambda c} p_{N+2} z^2 \sum_{n=0}^{\infty} \frac{(\lambda c z)^n}{n!} + \cdots$$
$$= \sum_{i=0}^{N} p_i e^{-\lambda c} e^{\lambda c z} + p_{N+1} z e^{-\lambda c} e^{\lambda c z} + \cdots + p_{N+k} z^k e^{-\lambda c} e^{\lambda c z} + \cdots$$
$$= \sum_{i=0}^{N} p_i e^{-\lambda c(1-z)} + e^{-\lambda c(1-z)} \left[\frac{P(z) - p_0 + p_1 z + \cdots + p_N z^N}{z^N} \right]$$

마지막 식 정리

$$P(z) e^{\lambda c(1-z)} = \sum_{i=0}^{N} p_i + \frac{P(z) - \sum_{n=0}^{N} p_n z^n}{z^N}$$

좌우변을 $P(z)$에 대해 정리

$$P(z)\left[z^N e^{\lambda c(1-z)} - 1\right] = \left(\sum_{i=0}^{N} p_t\right) z^N - \sum_{n=0}^{N} p_n z^n$$
$$\boldsymbol{P(z) = \frac{(\sum_{i=0}^{N} p_t) z^N - \sum_{n=0}^{N} p_n z^n}{z^N e^{\lambda c(1-z)} - 1}}$$

*대기행렬의 기초 2장(확률이론의 기초) 참고
확률생성함수: 확률변수 X에 대한 확률값들을 하나의 함수로 요약한 것임. 이산확률변수 X의 확률질량함수가 $p(k)$일 때, $G(z) = E[z^X] = \sum_{k=0}^{\infty} p(k) z^k$로 표현됨. 이는 z에 대한 함수로 확률변수 X의 정보를 담고 있어 X값 전부를 다루는 대신 z를 통해 쉽게 원하는 값을 유도함. 또한, 상태확률은 상태확률의 확률생성함수로 표시될 수 있음

B. $M/D/N@c$ 큐의 상태확률

- p_n의 확률생성함수(PGF, Probability Generating Function)(2/2)
  - $\boldsymbol{P(z) = \sum_{n=0}^{\infty} p_n z^n = \frac{(\sum_{i=0}^{N} p_t) z^N - \sum_{n=0}^{N} p_n z^n}{z^N e^{\lambda c(1-z)} - 1}}$

  - $N = 1$(고객이 1명)인 경우의 확률생성함수 구하는 방법

$\boldsymbol{P(z)}$ 공식 활용

$$P(z) = \frac{(\sum_{i=0}^{1} p_t) z^1 - \sum_{n=0}^{1} p_n z^n}{z^1 e^{\lambda c(1-z)} - 1} = \frac{p_0(z-1)}{z e^{\lambda c(1-z)} - 1}$$

$\boldsymbol{z = 1}$이면 분모와 분자가 0이 되므로, 로피탈의 정리 활용

$$P(1) = \lim_{z \to 1} \frac{p_0}{e^{\lambda c(1-z)} - \lambda c z e^{\lambda c(1-z)}} = \frac{p_0}{1 - \lambda c}$$

$\boldsymbol{z = 1}$이면, $\boldsymbol{P(1) = 1}$

$$\frac{p_0}{1 - \lambda c} = 1, \qquad \boldsymbol{p_0 = 1 - \lambda c}$$

$\boldsymbol{p_0}$을 대입

$$\boldsymbol{P(z) = \frac{(1 - \lambda c)(z-1)}{z e^{\lambda c(1-z)} - 1}}$$

***로피탈(l'Hopital)의 정리**

두 함수 $f(x)$와 $g(x)$가 $x = a$에서 모두 0으로 수렴하거나 모두 무한대로 발산하는 경우(즉, $\frac{0}{0}$ 또는 $\frac{\infty}{\infty}$ 꼴의 극한)에 $f(x)$와 $g(x)$가 $x = a$에서 미분 가능하다면, $\lim_{x \to a} \frac{f(x)}{g(x)} = \lim_{x \to a} \frac{f'(x)}{g'(x)}$가 성립함

B. $M/D/N@c$ 큐의 상태확률

- 시스템 내 평균 고객 수(Average Number of Customers in System)
  - $\boldsymbol{L = P'(z)|_{z=1} = P'(1)}$

증명#5

확률생성함수 $\boldsymbol{P(z)}$ 활용 $P(z) = \sum_{n=0}^{\infty} p_n z^n$

양변 $\boldsymbol{z}$로 미분 $P'(z) = \sum_{n=1}^{\infty} n p_n z^{n-1}$

$z = 1$ 대입 $P'(1) = \sum_{n=1}^{\infty} n p_n z^0 = E[n] = L$

B. $M/D/N@c$ 큐의 상태확률

*대기행렬의 기초 책 예제 4-7에 해당

예제 4-1

단위시간당 시스템으로 들어오는 고객의 도착률이 0.3이고, 어느 한순간에 고객은 한 명만 서비스를 받을 수 있으며, 서비스 주기인 c가 2인 $M/D/1@2$큐에 대하여 이 시스템의 상태를 나타내는 확률생성함수 $P(z)$를 구하고, 시스템 내에서 기다리고 있는 고객의 평균 수를 구하여라.

- 큐에서 기다리고 있는 평균 고객의 수: $L = P'(1)$
  - $P(z) = \frac{0.4(z-1)}{ze^{0.6(z-1)}-1}$
- 시스템 내 평균 고객 수
  - $P(z) = \frac{0.4(z-1)}{ze^{0.6(z-1)}-1} = \frac{0.4y}{(1+y)e^{-0.6y}-1} = \frac{0.4y}{(1+y)(1-0.6y+0.18y^2-\cdots)-1} = \frac{0.4y}{1-0.6y+0.18y^2+\cdots+y-0.6y^2+\cdots-1} = \frac{0.4y}{0.4y-0.42y^2+\cdots} \cong \frac{1}{1-1.05y}$
  - $P(z) \cong 1 + 1.05(z-1) + \cdots = 1.05z - 0.05 + \cdots \cong 1.05z - 0.05$

$\therefore P'(1) = 1.05$, 즉 시스템 내에서 기다리고 있는 고객의 평균 수는 1명

$M/X/\infty$ 큐

Ⅲ

A. 개요

- $M/X/\infty$ 큐
  - 고객 수보다 과도하게 많은 서버를 가지는 무한서버 시스템으로, 임의의 서비스 분포(X: M, G, D)를 가지는 대기행렬 시스템

구분	$M/M/\infty$ 큐	$M/G/\infty$ 큐	$M/D/\infty$ 큐
고객 도착과정	포아송분포		
고객 서비스과정	지수분포(M, Markov)	일반분포(G, General)	고정된 시간(D)
서버 수	무한(∞)		

- e.g., 고객 도착과정이 포아송분포를 따르고, 고객 서비스시간이 평균 3초인 $M/X/\infty$
  - 고객 4명(A, B, C, D)이 각각의 큐에서 동일한 시점에 도착함을 가정
  - 각 큐에서의 서비스시간: $M/M/\infty$ 큐(평균 3초인 지수분포), $M/G/\infty$ 큐(평균 3초, 표준편차 0.5초인 정규분포), $M/D/\infty$ 큐(고정 시간 3초)

고객	도착시점	$M/M/\infty$ 큐		$M/G/\infty$ 큐		$M/D/\infty$ 큐	
		서비스시간	종료시점	서비스시간	종료시점	서비스시간	종료시점
A	0초	1.1초	1.1초	2.9초	2.9초	3.0초	3.0초
B	2.7초	6.4초	9.1초	3.3초	6.0초	3.0초	5.7초
C	5.4초	2.3초	7.7초	2.5초	7.9초	3.0초	8.4초
D	9.1초	13.2초	22.3초	3.1초	12.2초	3.0초	12.1초

$M/M/\infty$
고객의 도착과정 / 고객의 서비스과정 / 서버의 수

B. $M/M/\infty$ 큐

- 정의
  - 고객 도착과정이 포아송분포를 따르고, 고객 서비스시간이 지수분포를 따르며 서버의 수와 버퍼의 제한이 없어 대기가 발생하지 않는 무한서버 대기행렬 시스템

- 특징
  - 고객이 도착하는 순서대로(FIFO, First-In First-Out) 서비스함
  - 시스템 서버는 무한하나, 고객이 거의 찾아오지 않는 경우 혹은 서버 수가 고객 수보다 충분히 큰 경우를 표현할 수 있음

- 켄달(Kendall) 표기방식
  - 고객 도착과정(Arrival Process): 포아송 분포(Possion Distribution, 표기 M)를 따름
    - λ: 고객의 단위시간(Unit Time)당 평균도착률(Average Arrival Rate)
  - 고객 서비스과정(Service Process): 지수분포(M: Markov)를 따름
    - μ: 하나의 서버에 의한 고객당 평균서비스율(Average Service Rate)
  - 서버의 수: 동시에 서비스가 가능한 서버의 수가 무한(∞)함
  - 버퍼의 크기: 고객이 기다리는 버퍼의 크기는 무한(∞)하여 표기상 생략됨

B. $M/M/\infty$ 큐

- 예시#1 – 초고속 네트워크
  - 회선의 용량이 초당 1기가(Giga=10^6)비트인 광케이블로 연결된 초고속 통신망에서, 최대속도 64 Kbps의 인터넷전화를 사용하는 네트워크
    - 회선 용량: 1 Gbps (1,000,000,000 bps)
    - 인터넷전화 1회선당 사용량: 최대 64 Kbps (64,000 bps)
    - 가능한 동시 접속 수: $\frac{1,000,000,000}{64,000} = 15,625$명 이상
  - 해당 네트워크에서는 제공 가능한 대역폭 용량에 비해 고객이 요구하는 소요대역이 작아지므로 전체 시스템으로 볼 때 서버가 무한히 많은 것처럼 작동하는 효과를 가짐

- 예시#2 – P2P(Peer-to-Pee) 서비스
  - P2P 서비스와 같이 백본 네트워크(Backbone Network)의 대역폭이 충분히 큰 광대역 네트워크에 접속한 다수의 P2P 사용자의 연결 요청이 발생하는 상황
    - 백본망 용량: 10 Gbps (10,000,000,000 bps)
    - P2P 사용자 1명당 평균 사용량: 1 Mbps (1,000,000 bps)
    - 가능한 동시 연결 수: $\frac{10,000,000,000}{1,000,000} = 10,000$명 이상
  - 해당 네트워크에서는 다수의 사용자가 개별적으로 접속/요청을 시도하더라도, 네트워크 대역폭이 충분히 크므로 전체 시스템으로 볼 때 무한한 병렬 처리가 가능한 서버 환경처럼 작동하는 효과를 가짐

B. $M/M/\infty$ 큐

- 파라미터 정의
  - 고객 평균도착률(Average Arrival Rate): $\boldsymbol{\lambda_k = \lambda\ (k = 1, 2, \dots)}$
  - 고객 평균서비스율(Average Service Rate): $\boldsymbol{\mu_k = k\mu\ (k = 1, 2, \dots)}$

- 시스템 상태확률(State Probability)
  - $\boldsymbol{p_k = \frac{\gamma^k}{k!} e^{-\gamma},\ k = 0, 1, \dots}$

증명#6

시스템 내 고객 수가 $k(k = 0, 1, \cdots)$명일 확률

$$p_k = \frac{\lambda_0 \lambda_1 \cdots \lambda_{k-1}}{\mu_1 \mu_2 \cdots \mu_k} p_0 = \frac{1}{k!} \left(\frac{\lambda}{\mu}\right)^k p_0$$

$\gamma = \frac{\lambda}{\mu}$ 활용 및 확률의 정의 (전체 확률의 합은 1)에 기반

$$\sum_{k=0}^{\infty} p_k = \sum_{k=0}^{\infty} \frac{\gamma^k}{k!} p_0 = 1$$

$\sum_{k=0}^{\infty} \frac{\gamma^k}{k!} = e^{\gamma}$ 활용

$$\sum_{k=0}^{\infty} \frac{\gamma^k}{k!} p_0 = 1, \qquad e^{\gamma} p_0 = 1, \qquad \boldsymbol{p_0 = e^{-\gamma}}$$

$p_0 = e^{-\gamma}$ 활용

$$p_k = \frac{\gamma^k}{k!} p_0 = \frac{\boldsymbol{\gamma^k}}{\boldsymbol{k!}} \boldsymbol{e^{-\gamma}}, \qquad \boldsymbol{k = 0, 1, \dots}$$

B. $M/M/\infty$ 큐

- 시스템 내 평균 고객 수(Average Number of Custormers in System)
  - $\boldsymbol{L = \gamma}$ 시스템 부하: $\gamma = \lambda/\mu$
- 고객의 시스템 내 평균체재시간(Mean Sojourn Time in System)
  - $\boldsymbol{W = \frac{1}{\mu}}$ 시스템에 고객 도착 시, 큐에서의 대기 없이 바로 서비스를 받을 수 있으므로 고객 대기시간이 없으며 서버에서의 서비스 시간만 존재함
  - $\boldsymbol{L = \lambda W}$ 관계가 성립함 (리틀의 공식)

증명#7

$$\boldsymbol{L = \gamma = \frac{\lambda}{\mu} = \lambda W}$$

시스템 부하 $\gamma = \lambda/\mu$ $W = \frac{1}{\mu}$

리틀의 공식(Little's Law)

$\boldsymbol{L = \lambda \times W}$

(시스템 내 평균 고객 수 = 평균도착률 × 평균체재시간)

**평균 고객 수=서비스 중인 고객 수+대기 중인 고객 수*

B. $M/M/\infty$ 큐

*대기행렬의 기초 책 예제 4-8에 해당

예제 4-2

통신시스템에서 트래픽이 가장 많이 발생하는 1시간을 최번시(Busy Hour)라고 한다. 최번시에서 1시간 동안 콜의 평균 도착률이 3이고, 콜당 평균지속시간이 180초라고 할 때 다음을 구하여라.

(1) 정상상태에서의 시스템 내 고객 수가 3명일 확률은 얼마인가?
(2) 정상상태에서의 시스템 내 고객 수는 평균 몇 명인가?
(3) 고객의 시스템 내 평균체재시간은 얼마인가?

- 시스템 부하: $\gamma = \frac{3\times180}{3600} = 0.15$

(1) 정상상태에서의 시스템 내 고객 수가 3명일 확률은 시스템 상태확률(p_k)에서 $k = 3$일 경우를 의미함
  - $p_k = \frac{\gamma^k}{k!}e^{-\gamma},\ p_3 = \frac{0.15^3}{3!}e^{-0.15} = 0.048\%$

(2) 정상상태에서의 시스템 내 평균 고객 수
  - $L = \gamma = 0.15$

(3) 고객의 시스템 내 평균체재시간
  - $W = \frac{L}{\lambda} = \frac{0.15}{3} \approx 0.05$

B. $M/M/\infty$ 큐

*대기행렬의 기초 책 문제 4-2에 해당

문제 4-1

어떤 네트워크에서 비디오 서비스를 제공하는데 최번시 1시간 동안 비디오 전화의 평균도착률이 2이고, 콜당 평균지속시간이 1,000초일때, 다음을 구하고 앞의 예제 4-8에서의 결과와 비교하라.

(1) 정상상태에서의 시스템 내의 고객의 수가 3명일 확률은 얼마인가?
(2) 정상상태에서의 시스템 내의 고객의 수는 평균 몇 명인가?
(3) 고객의 시스템 내 평균체재시간은 얼마인가?
(4) 앞의 예제 4-2에서의 결과와 비교하여 어떤 차이가 있는가?

- 시스템 부하: $\gamma = \frac{2\times1000}{3600} = 0.5556$

(1) 정상상태에서의 시스템 내 고객 수가 3명일 확률은 시스템 상태확률(p_k)에서 $k = 3$일 경우를 의미함
  - $p_k = \frac{\gamma^k}{k!}e^{-\gamma},\ p_3 = \frac{0.5556^3}{3!}e^{-0.5556} \approx 1.64\%$

(2) 정상상태에서의 시스템 내 평균 고객 수
  - $L = \gamma = 0.5556$

(3) 고객의 시스템 내에서의 평균체재시간
  - $W = \frac{L}{\lambda} = \frac{0.5556}{2} \approx 0.2778$초

(4) 비디오 서비스는 콜당 체재시간이 길어 시스템 내 고객 수와 시스템 부하가 예제 4-8의 서비스보다 상대적으로 많음

B. $M/M/\infty$ 큐

*대기행렬의 기초 책 문제 4-3에 해당

문제 4-2

어떤 네트워크에서 인터넷 서비스를 제공하는데 최번시 1시간 동안 평균 웹접속 요구의 수가 10개이고, 하나의 요구당 평균 지속시간이 30초라고 한다. 이때 다음의 물음에 답하라.

(1) 정상상태에서의 시스템 내의 고객의 수가 3명일 확률은 얼마인가?
(2) 정상상태에서의 시스템 내의 고객의 수는 평균 몇 명인가?
(3) 고객의 시스템 내 평균 체재시간은 얼마인가?
(4) 앞의 예제 4-2, 문제 4-1에서의 결과와 비교하여 어떤 차이가 있는가?

- 시스템 부하: $\gamma = \frac{10\times30}{3600} = 0.0833$

(1) 정상상태에서의 시스템 내 고객 수가 3명일 확률은 시스템 상태확률(p_k)에서 $k = 3$일 경우를 의미함
  - $p_k = \frac{\gamma^k}{k!}e^{-\gamma} = \frac{0.0833^3}{3!}e^{-0.0833} = 0.0089\%$

(2) 정상상태에서의 시스템 내 평균 고객 수
  - $L = \gamma = 0.0833$

(3) 고객의 시스템 내에서의 평균체재시간
  - $W = \frac{L}{\lambda} = \frac{0.0833}{10} \approx 0.00833$초

(4) 인터넷 서비스 중 웹 접속은 예제 4-8의 전화 서비스와 문제 4-2의 비디오 서비스에 비해 상대적으로 체재시간이 짧으며, 시스템 부하가 낮음

$M/G/\infty$
고객의 도착과정 / 고객의 서비스과정 / 서버의 수

C. $M/G/\infty$ 큐

- 정의
  - 고객 도착과정이 포아송분포를 따르고 고객 서비스시간이 일반분포를 따르며, 서버 수와 버퍼의 제한이 없어 대기가 발생하지 않는 무한서버 대기행렬 시스템

- 특징
  - 고객이 도착하는 순서대로(FIFO, First-In First-Out) 서비스함
  - 시스템 서버가 무한하고 다양한 서비스 시간 분포를 고려할 수 있으므로, 현실적인 처리 지연 등의 성능 예측에 유연함

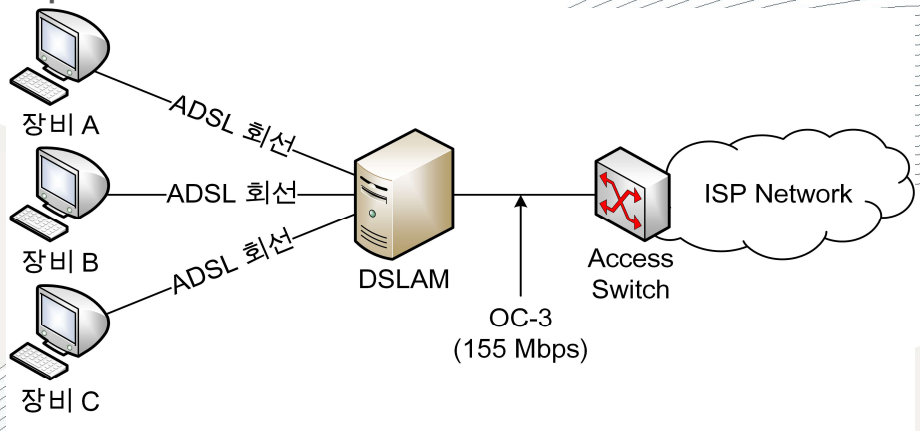
- 켄달(Kendall) 표기방식
  - 고객 도착과정(Arrival Process): 포아송 분포(Possion Distribution, 표기 M)를 따름
    - λ: 고객의 단위시간(Unit Time)당 평균도착률(Average Arrival Rate)
  - 고객 서비스과정(Service Process): 일반적인 분포(G: General)를 따름
    - μ: 고객당 평균서비스율(Average Service Rate) (즉, 평균서비스시간은 $\eta = 1/\mu$)
  - 서버의 수: 동시에 서비스가 가능한 서버의 수가 무한(∞)함
  - 버퍼의 크기: 고객이 기다리는 버퍼의 크기는 무한(∞)하여 표기상 생략됨

일반적인 분포(G: General)는 어떠한 분포도 포함할 수 있는 일반적인 분포로, 주로 정규분포, 균등분포, 파레토분포 등이 있음

C. $M/G/\infty$ 큐

*DSLAM: Digital Subscriber Line Access Multiplexer

- 예시 – ADSL(Asymmetric Digital Subscriber Line)
  - ADSL는 비동기식 디지털 가입자 회선 기술의 한 유형으로, 초당 수백 메가비트의 전송로에 가입자당 수 메가비트의 처리용량을 제공하는 가입자 회선 서비스임
  - 이는 하나의 네트워크 전송로에 다수의 가입자를 수용하는 것이 일반적이고, 다수의 고객을 DSLAM에서 다중화하여 고속 네트워크 접속 회선으로 수용함
    - 고객 도착과정(Arrival Process): 가입자가 네트워크로 전송하는 데이터 양이 일정 속도를 초과할 수 없고, 가입자 간에 상호 연관성이 없으므로 포아송분포를 따름
    - 고객 서비스과정(Service Process): 가입자의 네트워크 내 체재시간이 지수함수적으로 감소하지 않고 긴 시간부터 짧은 시간까지의 일반적인 분포를 따름
    - 서버의 수: 하나의 서버가 동시에 여러 명의 고객을 서비스하여야 하므로 서버가 가진 처리용량은 고객이 요구하는 서비스 처리용량보다 충분히 큼
  - 해당 서비스 환경에서는 네트워크 대역폭이 수백 Mbps에서 수 Gbps만큼 충분히 클 때, 가입자당 발생되는 트래픽 양이 수 Kbps 정도인 경우에 가입자 관점에서 서버가 무한한 모델로 간주할 수 있음



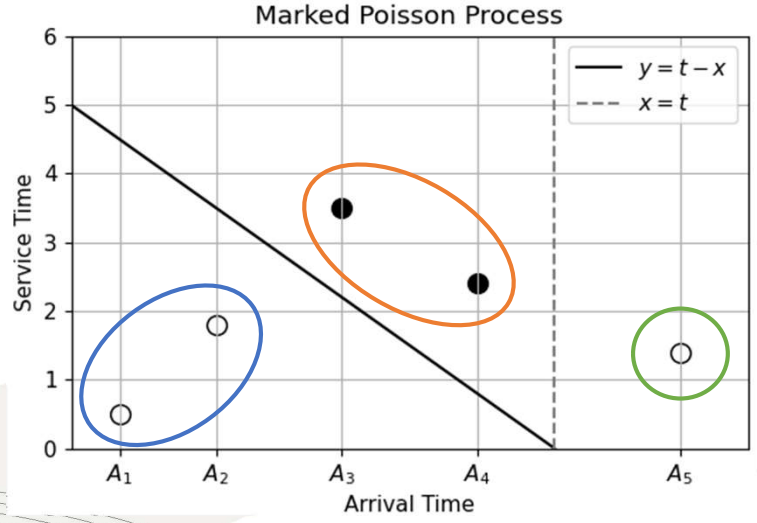
(그림 4) ADSL의 네트워크 접속 환경

C. $M/G/\infty$ 큐

- 파라미터 정의
  - 고객 평균도착률(Average Arrival Rate): $\boldsymbol{\lambda_k = \lambda\ (k = 1, 2, \dots)}$
  - 고객 평균서비스율(Average Service Rate): $\boldsymbol{\mu_k = k\mu\ (k = 1, 2, \dots)}$
  - 고객 도착시간: $\boldsymbol{A_n\ (n = 1, 2, \dots)}$
  - 고객 서비스시간: $\boldsymbol{V_n\ (n = 1, 2, \dots)}$
  - 시간 $\boldsymbol{t}$에서 시스템 내 고객 수: $\boldsymbol{Q(t)}$

$M/M/\infty$ 큐에서는 BDP(Birth and Death Process) 해석을 활용하여 시스템 상태확률(p_k)을 구함

- 시스템 상태확률(State Probability)(1/3)
  - MPP(Marked Poisson Process) 해석을 활용함
    - 포아송 과정에 속성(mark)을 추가하여 고객 도착사건이 가진 속성을 함께 고려함



(그림 5) MPP의 진행

- 주황색 부분: 시간 t 이내에 서비스 완료하지 못한 고객 (= 아직 서비스 중인 고객)
  ⇒ $A_k \le t$이고 $A_k + V_k > t$의 조건을 만족하는 고객 수
- 파란색 부분: 시간 t 기준으로 서비스 완료된 고객
  ⇒ $A_k + V_k < t$의 조건을 만족하는 고객 수
- 초록색 부분: 시간 t 기준으로 아직 도착하지 않은 고객
  ⇒ $A_k > t$의 조건을 만족하는 고객 수

$\therefore \boldsymbol{Q(t) = 2}$, (주황색 부분의 고객 수에 해당함)

C. $M/G/\infty$ 큐

- 시스템 상태확률(State Probability)(2/3)
  - 도착시간과 서비스시간인 (A, V)가 가질 수 있는 점 좌표(x, y)의 집합
    - $\boldsymbol{E_t = \{(x, y): 0 \le x \le t}$ 이고 $\boldsymbol{y > t - x\}}$
    - 시점 $\boldsymbol{t}$에서 시스템에 남아있는 경우는, 시점 $\boldsymbol{x}$가 시점 $\boldsymbol{t}$ 전에 도착했어야 하고, 서비스 시간 $\boldsymbol{V}(=$ 시점 $\boldsymbol{y})$가 $\boldsymbol{t - x}$보다 길어야 함
  - 집합 $\boldsymbol{E_t}$는 평균이 $E[E_t]$인 포아송분포를 따름
    - $\boldsymbol{E[E_t] = \lambda \int_0^t \{1 - G(t - x)\}\, dx}$

증명#8

집합 $\boldsymbol{E_t}$는 평균이 $E[E_t]$인 포아송분포를 따름

$$E[E_t] = \int_0^t \int_{t-x}^{\infty} \lambda(dx)\, dG(y) = \lambda \int_0^t \left\{ \int_{t-x}^{\infty} dG(y) \right\} dx$$

시간 x에 고객이 도착할 확률 — 서비스시간이 $t - x$보다 클 확률

내부적분은 $y = t - x$에서 시작하며, $G(y)$는 누적분포함수이므로

$$\int_{t-x}^{\infty} dG(y) = 1 - G(t - x)$$

$$E[E_t] = \lambda \int_0^t \{1 - G(t - x)\}\, dx$$

$\boldsymbol{G(y) = P\{V \le y\}}$: 서비스시간의 누적분포함수
$\boldsymbol{1 - G(t - x)}$: 시점 $\boldsymbol{x}$에 도착한 고객이 시점 $\boldsymbol{t}$까지 여전히 시스템에 머물 확률

C. $M/G/\infty$ 큐

- 시스템 상태확률(State Probability)(3/3)
  - 집합 E_t가 평균이 $E[E_t]$인 포아송분포를 따르므로 $Q(t)$가 k일 확률
    - $\boldsymbol{p_k = \lim_{t\to\infty} p_k(t) = \frac{(\lambda\eta)^k e^{-\lambda\eta}}{k!}, k = 0, 1, \cdots}$

고객 서비스시간에 대한 기대치는 $\boldsymbol{E[V] = \eta}$

증명#9

$Q(t) = k$일 확률

$$p_k(t) = P\{Q(t) = k\} = \frac{E[E_k]^k e^{-E[E_t]}}{k!}, \qquad k = 0, 1, \cdots$$

긴 시간($t \to \infty$) 동안의 $E[E_t]$

$$\lim_{t\to\infty} E[E_t] = \lambda \int_0^\infty \{1 - G(t - x)\} dx = \lambda\eta$$

$$\boldsymbol{p_k = \lim_{t\to\infty} p_k(t) = \frac{(\lambda\eta)^k e^{-\lambda\eta}}{k!}, \qquad k = 0, 1, \cdots}$$

$\gamma = \lambda\eta$인 경우

$$\boldsymbol{p_k = \frac{\gamma^k e^{-\gamma}}{k!}, \qquad k = 0, 1, \cdots}$$

*$M/G/\infty$ 큐에서의 시스템 내 고객이 k명 있을 확률(=시스템 상태확률)은 $M/M/\infty$ 큐에서의 시스템 내 고객이 k명 있을 확률과 동일함을 의미함

- $M/M/\infty$ 큐에서의 시스템 상태확률: $\boldsymbol{p_k = \frac{\gamma^k}{k!} e^{-\gamma}}$

C. $M/G/\infty$ 큐

- $M/M/\infty$ 큐와의 관계
  - 시스템 상태확률(=시스템 내 고객이 k명 있을 확률)이 동일함
    - 시스템 상태확률: $p_k = \frac{\gamma^k e^{-\gamma}}{k!},\ k = 0, 1, \cdots$
  - 시스템 상태확률을 통해 시스템 내 평균 고객 수 추정 가능하므로 해당 값도 동일함
    - 시스템 내 평균 고객 수: $L = \gamma$
  - 리틀의 공식에 따라 고객의 시스템 내 평균체재시간도 추정 가능하므로 해당 값도 동일함
    - 고객의 시스템 내 평균체재시간: $W = \frac{1}{\mu}$

C. $M/G/\infty$ 큐

한 지역에서 ADSL 서비스를 제공하는 통신사가 있다. 이 시스템은 다음과 같은 특성을 가진다.

- 고객의 데이터 세션 요청은 평균 도착률 $\lambda = 2$ 세션/분의 포아송 과정을 따름
- 각 세션의 데이터 사용 시간(=서비스 시간)은 평균 $\eta = 5$분이며, 분포는 일반 분포(G)를 따름
- 해당 시스템은 충분한 대역폭을 보유하고 있어 동시에 무한히 많은 고객을 수용할 수 있음

이때, 정상상태에서 시스템 내에 고객이 3명 있을 확률은?

- 정상상태에 대해서 시스템 내에 고객이 k명 있을 확률
  - $p_k = \lim_{t\to\infty} p_k(t) = \frac{(\lambda\eta)^k e^{-\lambda\eta}}{k!}, k = 0, 1, \cdots$

- $\lambda = 2, \eta = 5, \lambda\eta = 10, k = 3$

- $p_3 = \frac{10^3 \cdot e^{-10}}{3!} = \frac{1000 \cdot e^{-10}}{6} \approx \frac{1000 \cdot 4.53999 \times 10^{-5}}{6} \approx \frac{0.0454}{6} \approx 0.00757$

∴시스템 내에 고객 3명이 존재할 확률은 약 0.76%

$M/D/\infty$
고객의 도착과정 / 고객의 서비스과정 / 서버의 수

D. $M/D/\infty$ 큐

- 정의
  - 고객 도착과정이 포아송분포를 따르고 고객 서비스시간이 고정된 값을 가지며, 서버 수와 버퍼의 제한이 없어 대기가 발생하지 않는 무한서버 대기행렬 시스템

- 특징
  - 고객이 도착하는 순서대로(FIFO, First-In First-Out) 서비스함
  - 정해진 처리 시간이 요구되는 반복적이거나 실시간적인 작업 환경 모델링에 적합함

- 켄달(Kendall) 표기방식
  - 고객 도착과정(Arrival Process): 포아송 분포(Possion Distribution, 표기 M)를 따름
    - λ: 고객의 단위시간(Unit Time)당 평균도착률(Average Arrival Rate)
  - 고객 서비스과정(Service Process): 고정된 값(D: Deterministic)를 따름
    - μ: 고객당 평균서비스율 (d: 고정된 서비스시간)
  - 서버의 수: 동시에 서비스가 가능한 서버의 수가 무한(∞)함
  - 버퍼의 크기: 고객이 기다리는 버퍼의 크기는 무한(∞)하여 표기상 생략됨

D. $M/D/\infty$ 큐

- 예시 – 병원 예방접종 시스템
  - 시민들이 백신 접종소에 랜덤으로 도착하며, 각각의 접종 시간은 동일하게 2분인 예방접종 시스템
    - 평균적으로 시간당 30명의 시민이 도착함: $\lambda = 0.5$명/분
    - 간호사가 시민 1명을 정확히 2분간 접종함: 서비스시간 $= 2$분 (고정)
    - 접종 종료까지 시스템에 머무는 평균 시민 수: $0.5 \times 2 = 1$명
  - 해당 시스템에서는 고정된 시간만큼 접종을 진행하며, 대기 없이 진행될 수 있도록 수십 명의 간호사가 동시에 접종을 시행한다고 할 때, 간호사(서버)가 무한히 많아 보이는 효과를 가짐

D. $M/D/\infty$ 큐

- 파라미터 정의
  - 고객 평균도착률(Average Arrival Rate): $\boldsymbol{\lambda_k = \lambda\ (k = 1, 2, \dots)}$
  - 고객 평균서비스율(Average Service Rate): $\boldsymbol{\mu_k = k\mu\ (k = 1, 2, \dots)}$

- $M/M/\infty$ 큐, $M/G/\infty$ 큐와의 관계
  - 시스템 상태확률(=시스템 내 고객이 $\boldsymbol{k}$명 있을 확률)이 동일함
    - 시스템 상태확률: $\boldsymbol{p_k = \frac{\gamma^k e^{-\gamma}}{k!},\ k = 0, 1, \cdots}$
  - 시스템 상태확률을 통해 시스템 내 평균 고객 수 추정 가능하므로 해당 값도 동일함
    - 시스템 내 평균 고객 수: $\boldsymbol{L = \gamma}$
  - 리틀의 공식에 따라 고객의 시스템 내 평균체재시간도 추정 가능하므로 해당 값도 동일함
    - 고객의 시스템 내 평균체재시간: $\boldsymbol{W = \frac{1}{\mu}}$

추가예제

IV

추가예제 1(1/2)

한 OTT 플랫폼은 다중 채널 기반의 비디오 스트림 처리를 지원한다. 사용자가 고화질 콘텐츠를 시청할 경우, 한 번의 요청으로 2개 이상의 비디오 채널이 동시에 점유된다. 이 시스템에서 사용자 스트리밍 요청은 초당 평균 3건 발생하며, 이는 집단 도착(Batch Arrival)으로 들어온다. 또한 각 요청은 각 일괄 크기(Batch Size) $k \in \{1,2,3\}$에 대한 채널을 점유한다: $q_1 = 0.2, q_2 = 0.5, q_3 = 0.3$. 해당 시스템의 전체 서버는 5개이고, 각 서버는 단위시간 당 평균 $\mu = 1$의 속도로 요청을 처리할 수 있으며, 버퍼는 없다. 이때 시스템의 상태 확률이 $p_3 = 0.3, p_4 = 0.2, p_5 = 0.1$로 관찰되었을 때, 각 일괄 크기(Batch Size) $k \in \{1,2,3\}$에 대한 채널 요청 도착률 λ_k를 계산하라. 또한, 위 상태확률이 주어졌을 때의 브러킹확률(Blocking Probability)을 계산하라.

- 시스템 유형: $M^X/M/c/c$ 큐 시스템
- 서버 수: $c = 5$
- 서버 하나의 단위시간 당 평균서비스율: $\mu = 1$
- 각 요청의 평균도착률: $\lambda = 3$
- 일괄 크기(Batch Size) 분포: $q_1 = 0.2, q_2 = 0.5, q_3 = 0.3$
- 시스템 상태확률: $p_3 = 0.3, p_4 = 0.2, p_5 = 0.1$
- 각 일괄 크기(Batch Size) $k \in \{1,2,3\}$에 대한 채널 요청 도착률($\lambda_k = \lambda \cdot q_k$)
  - $\lambda_1 = 3 \cdot 0.2 = 0.6$
  - $\lambda_2 = 3 \cdot 0.5 = 1.5$
  - $\lambda_3 = 3 \cdot 0.3 = 0.9$

$\therefore$ **채널 요청 도착률은 각각 $\lambda_1 = 0.6, \lambda_2 = 1.5, \lambda_3 = 0.9$임**

추가예제 1(2/2)

한 OTT 플랫폼은 다중 채널 기반의 비디오 스트림 처리를 지원한다. 사용자가 고화질 콘텐츠를 시청할 경우, 한 번의 요청으로 2개 이상의 비디오 채널이 동시에 점유된다. 이 시스템에서 사용자 스트리밍 요청은 초당 평균 3건 발생하며, 이는 집단 도착(Batch Arrival)으로 들어온다. 또한 각 요청은 다음 중 하나의 채널 수를 점유한다: $q_1 = 0.2, q_2 = 0.5, q_3 = 0.3$. 해당 시스템의 전체 서버는 5개이고, 각 서버는 단위시간 당 평균 $\mu = 1$의 속도로 요청을 처리할 수 있으며, 버퍼는 없다. 이때 시스템의 상태 확률이 $p_3 = 0.3, p_4 = 0.2, p_5 = 0.1$로 관찰되었을 때, 각 일괄 크기(Batch Size) $k \in \{1,2,3\}$에 대하여 단위시간 당 채널 요청 도착률인 λ_k를 계산하라. 또한, 위 상태확률이 주어졌을 때의 브러킹확률(Blocking Probability)을 계산하라.

- 브러킹확률은 각 일괄 크기(Batch Size) $k \in \{1,2,3\}$에 대한 요청이 들어올 때, 시스템이 $c - k + 1$개보다 많은 서버를 점유 중이라면 해당 요청은 차단(Block)됨
- $k \in \{1,2,3\}$ 이라서 $K = 3$이므로, 브러킹(Blocking) 조건은 현재 채널 점유 수 ≥ 3, 즉 $p_3 + p_4 + p_5$을 구하면 됨
- 따라서, 브러킹확률을 구하면 다음과 같음
  - $\Omega = \frac{1}{\lambda}\sum_{i=0}^{K-1}\left(p_{c-i}\sum_{k=i+1}^{K}\lambda_k\right) = \frac{1}{3}\left(p_5 \cdot \sum_{k=1}^{3}\lambda_k + p_4 \cdot \sum_{k=2}^{3}\lambda_k + p_3 \cdot \sum_{k=3}^{3}\lambda_k\right) = \frac{1}{3}(0.3 + 0.48 + 0.27) = \frac{1.05}{3} = 0.35$

∴ **브러킹확률은 약 35%임**

추가예제 3(1/2)

한 통합 네트워크 서버는 VoIP 통화 스트림과 CCTV 영상 스트림을 동시에 처리한다. 이 시스템은 서버 수가 제한되어 있고, 요청 도착은 일괄 도착(Batch Arrival) 형태로 이루어지며, 대기열은 존재하지 않는다. 각각의 요청 중 VoIP 요청은 한 번에 1개의 채널만 점유하며 초당 평균 2건이 도착하고, CCTV 요청은 한 번에 3개의 채널을 점유하며 초당 평균 0.5건이 도착한다. 서버는 총 6개이며, 각 서버는 단위시간 당 평균적으로 1건의 요청을 처리할 수 있다. 이때, VoIP와 CCTV 각각의 요청에 대한 브러킹(Blocking)이 발생하는 상태 조합을 설명하라. 또한, 시스템 상태확률이 $p_4 = 0.15, p_5 = 0.25, p_6 = 0.35$일 때 각 요청의 브러킹확률을 계산하고, 상태가 p_6일 때 브러킹확률을 줄이기 위한 방법을 설명하라.

- 시스템 유형: $M^X/M/c/c$ 큐 시스템
- 서버 수: $c = 6$, 시스템 상태확률: $p_4 = 0.15, p_5 = 0.25, p_6 = 0.35$
- 서버 하나의 단위시간 당 평균서비스율: $\mu = 1$
- 각 일괄 크기(Batch Size) k에 대한 채널 요청 도착률
  - VoIP: 채널 수 $k = 1$, 도착률 $\lambda_1 = 2$
  - CCTV: 채널 수 $k = 3$, 도착률 $\lambda_3 = 0.5$
- 브러킹확률은 각 일괄 크기(Batch Size) $k \in \{1,3\}$에 대한 요청이 들어올 때, 시스템이 $c - k + 1$개보다 많은 서버를 점유 중이라면 해당 요청은 차단(Block)됨
  - VoIP($k = 1$): 브러킹조건: 현재 점유 채널수 ≥ 6
  - CCTV($k = 3$): 브러킹조건: 현재 점유 채널수 ≥ 4
- 따라서, 브러킹확률을 구하면 다음과 같음: $\Omega = \frac{1}{\lambda}\sum_{i=0}^{K-1}\left(p_{c-i}\sum_{k=i+1}^{K}\lambda_k\right) = \frac{1}{2.5}\sum_{i=0}^{2}\left(p_{6-i}\sum_{k=i+1}^{3}\lambda_k\right) = \frac{1}{2.5}(p_6 \cdot (\lambda_1 + \lambda_3) + p_5 \cdot (\lambda_3) + p_4 \cdot (\lambda_3)) = \frac{1}{2.5}(0.875 + 0.125 + 0.075) = 0.43$

∴ 브러킹확률은 약 43%임

추가예제 3(2/2)

한 통합 네트워크 서버는 VoIP 통화 스트림과 CCTV 영상 스트림을 동시에 처리한다. 이 시스템은 서버 수가 제한되어 있고, 요청 도착은 일괄 도착(Batch Arrival) 형태로 이루어지며, 대기열은 존재하지 않는다. 각각의 요청 중 VoIP 요청은 한 번에 1개의 채널만 점유하며 초당 평균 2건이 도착하고, CCTV 요청은 한 번에 3개의 채널을 점유하며 초당 평균 0.5건이 도착한다. 서버는 총 6개이며, 각 서버는 단위시간 당 평균적으로 1건의 요청을 처리할 수 있다. 이때, VoIP와 CCTV 각각의 요청에 대한 브러킹(Blocking)이 발생하는 상태 조합을 설명하라. 또한, 시스템 상태확률이 $p_4 = 0.15, p_5 = 0.25, p_6 = 0.35$일 때 각 요청의 브러킹확률을 계산하고, 상태가 p_6일 때 브러킹확률을 줄이기 위한 방법을 설명하라.

- 시스템 상태가 6일 때, 브러킹확률을 줄이기 위한 방법은 다음과 같음
  1. 서버 수(c) 증설
     - 현재 $c = 6$에서, 7이나 8로 증가시킴으로써 포화 상태의 빈도를 감소시킴
  2. CCTV 영상 스트림의 점유 채널(k) 조정
     - 점유하고 있는 채널을 3개 이하로 감소시킴
     - 브러킹조건(현재 채널 3개 점유): $c - k + 1 = 6 - 3 + 1 = 4$
     - 브러킹조건(감소하여 채널 2개 점유): $c - k + 1 = 6 - 2 + 1 = 5$ => 효율이 높아지는 효과
  3. CCTV 요청도착률(λ_3) 조정
     - CCTV는 채널을 많이 점유함에 따라 요청 도착률이 브러킹 조건에 가장 큰 영향을 미침
     - 브러킹확률 수식에는 요청 도착률이 $i + 1$부터 K까지 누적합으로 들어가므로 λ_3의 값을 줄이면 브러킹확률 값도 감소하게 됨

추가예제 2

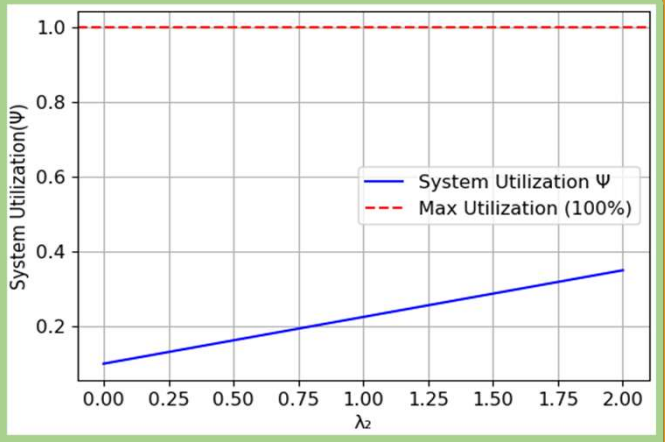
한 보안관제 시스템은 IDS 로그를 병렬로 처리하기 위해 다중 서버 기반의 로그 처리 구조를 채택하고 있다. 각 로그 그룹은 최대 2개의 서버를 동시에 점유할 수 있고, 한 그룹은 병렬적으로 파싱 및 디코딩함에 따라 1개 또는 2개의 서버를 동시에 사용한다. 해당 시스템은 총 4개의 서버를 가지고 있으며 버퍼는 없고, 일괄 도착(Batch Arrival) 방식으로 요청이 들어온다. 또한, 각 서버의 처리율은 $\mu = 2$이고, 서버 점유 수에 따른 요청 도착률이 다음과 같다: $\lambda_1 = 1$ (서버 1개 점유), $\lambda_2 = 0.5$ (서버 2개 점유). 이 시스템의 상태확률이 $p_3 = 0.3, p_4 = 0.2$로 주어질 때 이 시스템의 서버 자원 효용을 계산하라. 또한, 시스템 효용이 낮은 경우, λ_2를 점진적으로 증가시켰을 때 효용에 어떤 변화가 생기는지 그래프와 함께 분석하라.

- 시스템 유형: $M^X/M/c/c$ 큐 시스템
- 서버 수: $c = 4$
- 서버 하나의 단위시간 당 평균서비스율: $\mu = 2$
- 시스템 상태확률: $p_3 = 0.3, p_4 = 0.2$
- $k \in \{1,2\}$ 이므로 $K = 2$
- 각 일괄 크기(Batch Size) $k \in \{1,2\}$에 대한 채널 요청 도착률: $\lambda_1 = 1,\ \lambda_2 = 0.5$
- 시스템 효용: $\Psi = \frac{\sum_{k=1}^{K}[k \times \lambda_k \times (1 - \sum_{i=c-k+1}^{c} p_i)]}{c\mu}$

  $= \frac{1 \times \lambda_1 \times (1-p_4) + 2 \times \lambda_2 \times (1-p_3-p_4)}{4 \times 2} = 0.1625$

∴ 시스템 효용은 0.1625

- 시스템 효용은 낮을수록 사용자의 서버 점유율이 낮은 것을 의미함
- λ_2를 증가시킬수록, 병렬 처리가 필요한 요청(서버 2개 점유)의 비중이 증가하고, 시스템 자원 사용률이 선형적 증가함
- 만약 시스템 효용이 낮은 경우, 서버의 자원 낭비 우려가 있어 λ_2를 증가시키기도 함
- 또는 처리 지연이 발생하는 경우, 서버 2개를 통한 병렬 처리 비중을 늘리기 위해 λ_2를 증가시키기도 함



(그림 6) λ_2 증가에 따른 시스템 효용 분석 그래프

IV 4 추가예제

추가예제 4

한 보안 로그 스케줄러는 네트워크 보안 이벤트 발생 시 로그를 수집하여 주기적으로 처리하는 시스템으로 구성된다. 해당 시스템에서 보안 이벤트는 포아송 도착을 따르며, 평균적으로 초당 3건이 도착한다. 해당 시스템은 10초마다 주기적으로 동작하며, 최대 5개의 로그를 한 번에 처리할 수 있다. 이 시스템이 한 번에 최대 1개의 로그만 처리할 수 있다고 가정할 때, 해당 주기 내에서 로그의 개수에 대한 확률생성함수 $P(z)$를 유도하라. 또한, 해당 시스템이 안정적인지 판단하라.

- 시스템 유형: $M/D/N@c$ 큐 시스템
- 평균 도착률: $\lambda = 3$
- 주기: $c = 10$초마다 처리
- 주기 동안 도착하는 평균 로그 수: $\lambda \cdot c = 3 \times 10 = 30$
- 최대 1개의 로그만 처리 가능하다고 가정하면, $N = 1$로 간주하여 확률생성함수를 유도할 수 있음
- $P(z) = \sum_{n=0}^{\infty} p_n z^n = \frac{(\sum_{i=0}^{1} p_t)z^1 - \sum_{n=0}^{1} p_n z^n}{z^1 e^{\lambda c(1-z)} - 1} = \frac{p_0(z-1)}{ze^{\lambda c(1-z)} - 1}$ 에서 $z = 1$ 이면 분모와 분자가 0이 되므로 로피탈의 정리를 이용함
- $P(1) = \lim_{z \to 1} \frac{p_0}{e^{\lambda c(1-z)} - \lambda c z e^{\lambda c(1-z)}} = \frac{p_0}{1-\lambda c}$, 여기서 $z = 1$이면 $P(1) = 1$이 되므로 $\frac{p_0}{1-\lambda c} = 1$, 즉 $p_0 = 1 - \lambda c$
- 이 p_0을 대입하면 확률생성함수는 $P(z) = \frac{(1-\lambda c)(z-1)}{ze^{\lambda c(1-z)} - 1}$이고, 여기에 $\lambda = 3, c = 10$을 대입하면 최종적으로 다음과 같음
  : $P(z) = \frac{-2 \cdot (z-1)}{ze^{30(1-z)} - 1}$
- 시스템 안정성을 판단하기 위해서는 $\lambda c < N$ 인지 확인해야 함
- 현재 처리량은 $N = 1$이므로, $\lambda c = 30 > N$이어서 안정적으로 보기 어려우며, 이를 안정적으로 만들기 위해서는 주기(c)를 줄여서 더 자주 처리하게 하거나, 처리량(N)을 증가시켜서 한 번에 여러 개를 처리하는 등의 방법이 있음

∴ 확률생성함수는 $P(z) = \frac{-29 \cdot (z-1)}{ze^{30(1-z)} - 1}$, 시스템은 안정적으로 보기 어려운 상황

추가예제 5(1/2)

한 정책 기반 패킷 처리 시스템은 네트워크 트래픽으로부터 발생하는 이벤트를 수집한 후, 일정 시간 간격으로 처리하는 구조이다. 해당 시스템에서 보안 이벤트는 포아송분포를 따르며 평균적으로 초당 0.4건이 도착한다. 시스템은 5초마다 주기적으로 작동하며, 매 주기마다 최대 3개의 패킷을 동시에 처리할 수 있다. 또한, 이 시스템은 혼잡 방지를 위한 정책 기반 드롭(drop) 기능이 포함되며, 이 드롭 정책은 매 주기마다 큐에 누적된 패킷 수가 4개 이상일 경우, 해당 시점의 초과 패킷은 폐기된다고 정의된다. 이때 이 시스템에서의 평균 큐 길이 $L = P'(1)$를 구하라. 또한, 평균적으로 얼마나 자주 드롭 정책이 실행되는지 그래프와 함께 분석하라.

- 시스템 유형: $M/D/N@c$ 큐 시스템
- 평균 도착률: $\lambda = 0.4$
- 주기: $c = 5$초마다 처리
- 주기 동안 도착하는 평균 로그 수: $\lambda \cdot c = 0.4 \times 5 = 2$
- 1회 주기 최대 처리량: $N = 3$

*멱급수 근사식:

$$e^{-x} \approx 1 - x + \frac{x^2}{2!} - \frac{x^3}{3!} + \frac{x^4}{4!} - \cdots$$

- 평균 큐 길이는 $L = P'(1)$와 확률생성함수를 활용하여 계산 가능함: $P(z) = \frac{(1-\lambda c)(z-1)}{ze^{\lambda c(1-z)}-1} = \frac{(-1)(z-1)}{ze^{2(z-1)}-1}$
- 이 식은 $z = 1$일 때 분모가 0이 되므로, 미분 대신에 $P(z)$에 대해 $z - 1 = y$로 치환 후, 지수함수 멱급수 전개식을 통해 근사적으로 계산해야 함
  - $P(z) = \frac{(-1)(z-1)}{ze^{2(z-1)}-1} = \frac{-y}{(1+y)e^{-2}-1} = \frac{-y}{(1+y)\left(1-2y+\frac{(2y)^2}{2!}-\frac{(2y)^3}{3!}+\cdots\right)-1} \cong \frac{1}{1-\frac{2}{3}y^2+\cdots}$ 이 값에서 치환한 부분을 원래대로 계산하면,

$P(z) \cong 1 + \frac{2}{3}(z-1)^2 + \cdots \cong 0$에 근사함

∴ 평균 큐 길이가 0에 근사하므로, 거의 모든 경우에 큐에서 대기하는 고객이 거의 없음을 의미함. 이는 매 주기마다 평균적으로 2개의 로그가 도착하고 3개까지도 처리 가능하므로 버퍼가 찰 일이 거의 없음을 의미함

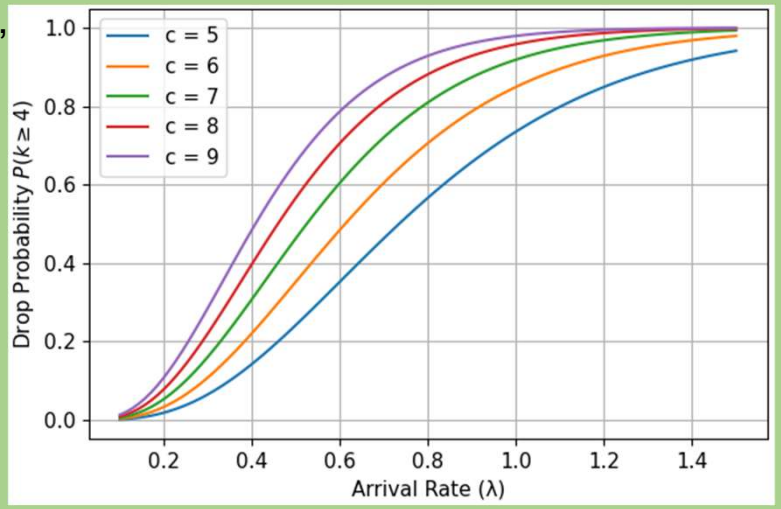
추가예제 5(2/2)

한 정책 기반 패킷 처리 시스템은 네트워크 트래픽으로부터 발생하는 이벤트를 수집한 후, 일정 시간 간격으로 처리하는 구조이다. 해당 시스템에서 보안 이벤트는 포아송분포를 따르며 평균적으로 초당 0.4건이 도착한다. 시스템은 5초마다 주기적으로 작동하며, 매 주기마다 최대 3개의 패킷을 동시에 처리할 수 있다. 또한, 이 시스템은 혼잡 방지를 위한 정책 기반 드롭(drop) 기능이 포함되며, 이 드롭 정책은 매 주기마다 큐에 누적된 패킷 수가 4개 이상일 경우, 해당 시점의 초과 패킷은 폐기된다고 정의된다. 이때 이 시스템에서의 평균 큐 길이 $L = P'(1)$를 구하라. 또한, 평균적으로 얼마나 자주 드롭 정책이 실행되는지 그래프와 함께 분석하라.

- 매 주기마다 큐에 누적 패킷 수가 4개 이상인 경우 드롭 정책이 실행되는데, 평균적으로 이 정책이 실행될 확률은 $P(N \geq 4)$로 표현할 수 있으며,

  시스템 상태확률 공식 $p_k = \frac{\gamma^k e^{-\gamma}}{k!}$ 을 활용하여 다음과 같이 구할 수 있음
  - 현재 시스템 부하: $\lambda \cdot c = 0.4 \times 5 = 2$
  - $P(N \geq 4) = 1 - (p_0 + p_1 + p_2 + p_3) = 1 - e^{-2}\left(1 + 2 + 2 + \frac{4}{3}\right) \approx 0.143$

- 드롭 정책 실행 확률은 동일한 평균도착률($\lambda = 0.4$)에서도 주기 시간(c)이 증가할수록 더 자주 실행됨
- 또한, x축에서와 같이 평균도착률(λ)이 증가할수록 드롭 정책 실행 확률이 증가하는 것을 볼 수 있음



(그림 7) 드롭 정책 실행 확률 비교 그래프

추가예제 6(1/2)

한 네트워크 처리 시스템은 외부에서 발생하는 보안 이벤트를 주기마다 일괄적으로 처리하는 구조를 가진다. 이때 이벤트는 포아송분포를 따르며 평균 도착률은 초당 $\lambda = 0.25$이다. 시스템은 주기마다 한 번씩 작동하며, 매 주기마다 최대 2개의 이벤트만 동시에 처리할 수 있다. 이때 시스템 설계를 위해 다음의 두 가지 스케줄 주기 옵션을 비교하고자 한다:

- 시스템 A는 10초마다 한 번 이벤트를 처리하는 구조
- 시스템 B는 5초마다 한 번 이벤트를 처리하는 구조

두 시스템은 동일한 처리량 조건을 가지며 주기만 다르다고 할 때, 각 시스템에 대한 평균 대기 고객 수를 구하고, 어떤 시스템이 더 효율적인지 그래프와 함께 분석하라.

- 시스템 유형: $M/D/N@c$ 큐 시스템
- 평균 도착률: $\lambda = 0.25$
- 1회 주기 최대 처리량: $N = 2$ (매 주기당 최대 2개의 이벤트 처리 가능)
- 주기: 시스템 A는 $c = 10$초, 시스템 B는 $c = 5$초
- 주기 동안 도착하는 평균 이벤트 수: 시스템 A는 $\lambda \cdot c = 0.25 \times 10 = 2.5$, 시스템 B는 $\lambda \cdot c = 0.25 \times 5 = 1.25$
- 평균 대기 고객 수는 $L = P'(1)$와 확률생성함수를 활용하여 계산 가능함: $P(z) = \frac{(1-\lambda c)(z-1)}{ze^{\lambda c(1-z)}-1}$
- 이 식은 $z = 1$일 때 분모가 0이 되므로, 미분 대신에 $P(z)$에 대해 $z - 1 = y$로 치환 후, 지수함수 멱급수 전개식을 통해 근사적으로 계산해야 함
  - 시스템 A의 평균 대기 고객 수: $L_A = P(z) = \frac{(1-2.5)(z-1)}{ze^{2.5(z-1)}-1} \cong 0.869$에 근사함
  - 시스템 B의 평균 대기 고객 수: $L_B = P(z) = \frac{(1-1.25)(z-1)}{ze^{1.25(z-1)}-1} \cong 0.181$에 근사함

∴ 시스템 A과 시스템 B의 평균 대기 고객 수는 모두 0명에 가까움

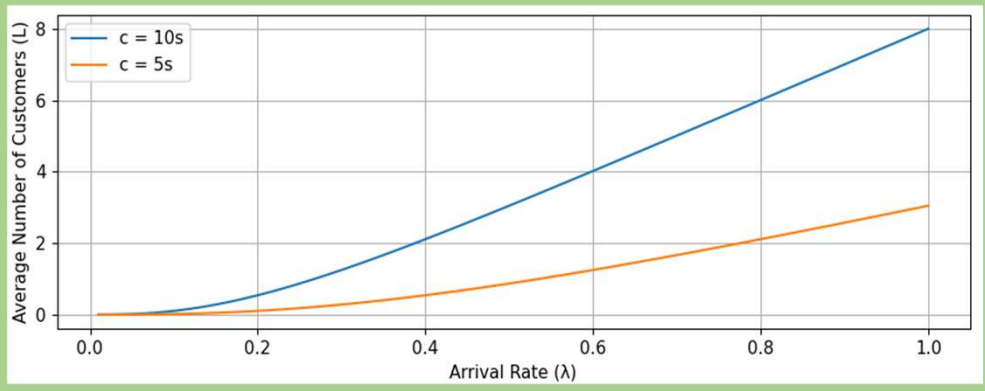
추가예제 6(2/2)

한 네트워크 처리 시스템은 외부에서 발생하는 보안 이벤트를 주기마다 일괄적으로 처리하는 구조를 가진다. 이때 이벤트는 포아송분포를 따르며 평균 도착률은 초당 $\lambda = 0.25$이다. 시스템은 주기마다 한 번씩 작동하며, 매 주기마다 최대 2개의 이벤트만 동시에 처리할 수 있다. 이때 시스템 설계를 위해 다음의 두 가지 스케줄 주기 옵션을 비교하고자 한다:

- 시스템 A는 10초마다 한 번 이벤트를 처리하는 구조
- 시스템 B는 5초마다 한 번 이벤트를 처리하는 구조

두 시스템은 동일한 처리량 조건을 가지며 주기만 다르다고 할 때, 각 시스템에 대한 평균 대기 고객 수를 구하고, 어떤 시스템이 더 효율적인지 그래프와 함께 분석하라.

- 두 시스템의 주기 시간, 도착률에 기반하여 그래프를 도출한 결과는 다음과 같음
- 평균도착률이 0에 근사할 정도로 충분히 작을수록 짧은 주기를 가진 시스템에서는 불필요하게 과도한 작동이 이루어지게 되므로 시스템 A를 활용하는 것이 더 효율적임



(그림 8) 시스템 A, B의 평균 대기 고객 수 비교 그래프

- 추가예제 6의 기존 파라미터에 의하면, 시스템 A와 B 모두 대기 고객이 거의 없었으나, 시스템 B에서의 평균 대기 고객 수가 더 작은 값을 가지는 것으로 계산됨
- 따라서, 현재 시스템에서는 시스템 B가 A보다 더 효율적인 것으로 해석할 수 있음

추가예제 7

한 온라인 사용자 인증 시스템은 인증 요청을 병렬로 처리하는 구조로 되어 있으며, 각 요청은 외부 MFA 확인 및 감사 로그 저장 등의 과정을 포함한다. 이 시스템은 무한한 수의 서버를 보유하며, 모든 요청이 즉시 처리된다. 사용자 인증 요청은 포아송분포를 따르며, 평균적으로 초당 20건이 도착한다. 또한, 각 인증 처리 시간은 평균 1.8초가 소요되며, 서비스 시간 분포는 임의의 일반 분포를 따른다고 가정한다. 이때 이 시스템의 부하를 계산하고, 시스템 내에 동시에 2명 이상 존재할 확률을 구하라.

- 시스템 유형: $M/G/\infty$ 큐 시스템
- 평균 도착률: $\lambda = 20$요청/초
- 평균서비스시간: $\frac{1}{\mu} = 1.8$초
- 시스템 부하: $\gamma = \frac{\lambda}{\mu} = 20 \times 1.8 = 36$
- 시스템 내 동시에 2명 이상 존재할 확률은 $P(N \geq 2)$로 표현할 수 있으며, 시스템 상태확률 공식 $p_k = \frac{\gamma^k e^{-\gamma}}{k!}$ 을 활용하여 다음과 같이 구할 수 있음
  - $P(N \geq 2) = 1 - (p_0 + p_1) = 1 - \left(\frac{36^0 e^{-36}}{0!} + \frac{36^1 e^{-36}}{1!}\right) = 1 - e^{-3}\ (1 + 36) = 1 - 8.5822 \times 10^{-15} \approx 1$

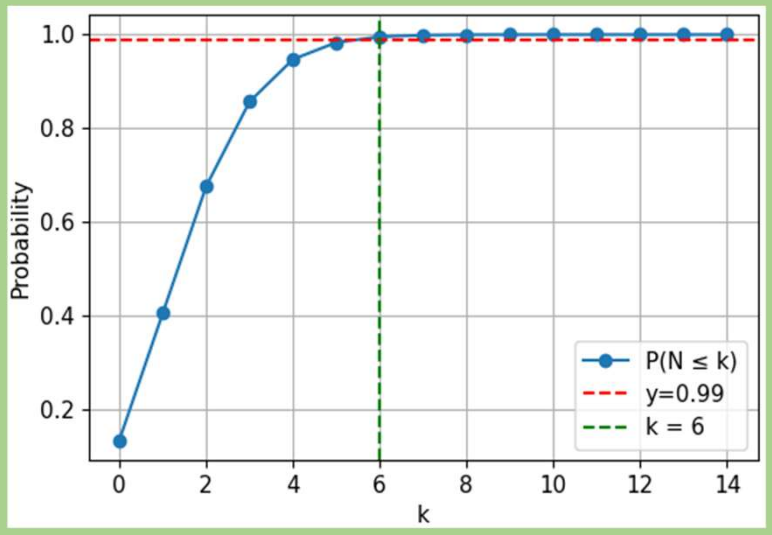
∴ 시스템 부하가 36으로 매우 큰 시스템임에 따라, 시스템 내에 2명 이상 존재할 확률은 거의 100%에 가까움

추가예제 8

한 실시간 피싱 탐지 처리 서버는 인터넷 트래픽에서 발생하는 탐지 요청을 수집하고, 이를 병렬로 분석하는 구조로 설계되어 있다. 해당 시스템은 요청 도착을 포아송분포로 따르며, 평균적으로 초당 5건의 탐지 요청이 발생한다. 또한, 각 요청은 평균적으로 0.4초의 처리가 소요되며, 요청 간 처리 시간은 일반적인 분포를 따르고, 해당 시스템은 무한한 병렬 서버를 통해 모든 요청을 즉시 처리할 수 있다고 가정한다. 이때 시스템 설계자는 시스템 내 탐지 요청 수가 k개를 초과하지 않을 확률이 99% 이상이 되도록 하기 위해, 최소 몇 개의 요청까지 동시 처리를 보장해야 하는지 계산하라.

- 시스템 유형: $M/G/\infty$ 큐 시스템
- 평균 도착률: $\lambda = 5$요청/초
- 평균서비스시간: $\frac{1}{\mu} = 0.4$초
- 시스템 부하: $\gamma = \frac{\lambda}{\mu} = 5 \times 0.4 = 2$
- 시스템 내 탐지 요청 수가 k개를 초과하지 않을 확률이 99% 이상인 것은 $P(N \le k) \ge 0.99$인 k의 최소값 구하기를 의미함
  - $P(N \le k) = \sum_{n=0}^{k} \frac{2^n \cdot e^{-2}}{n!} \ge 0.99$
- 그래프에서와 같이 k가 6인 경우에 99%가 되므로, 최소 6개의 요청까지 동시 처리를 보장해야 함을 분석할 수 있음

∴ 최소 6개의 탐지 요청까지 동시 처리를 보장해야 함

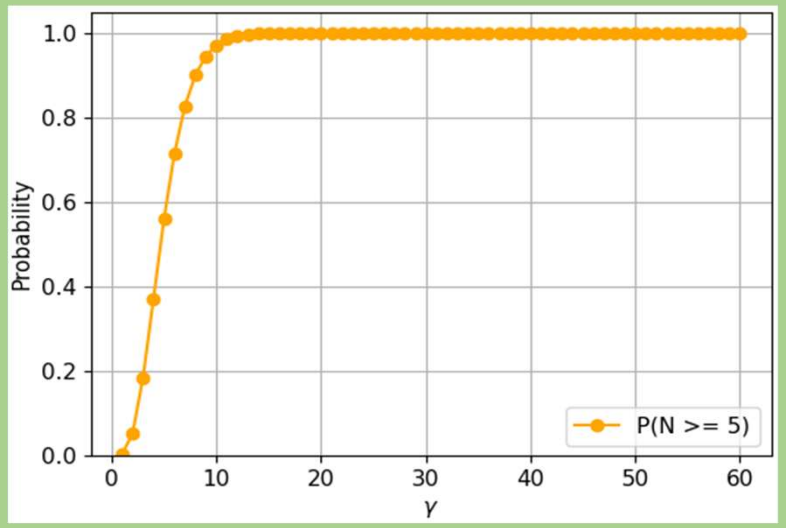


(그림 9) 동시 처리 가능한
시스템 내 탐지 요청 수(k) 그래프

추가예제 9

한 보안 분석 시스템은 인터넷 트래픽 상에서 유입되는 악성 URL을 실시간으로 탐지하고 분석하는 구조로 설계되어 있다. 이 시스템에서 분석 요청은 포아송 도착과정을 따르며, 각 URL 분석은 무한한 서버를 통해 병렬 처리된다. 각 서버는 평균적으로 2초가 소요되고 분석 시간은 정규분포 $N(\mu = 2초, \sigma^2 = 0.25)$를 따른다. 해당 시스템의 정상 상태(Steady-State)에서 1분 동안 관찰하였을 때 평균적으로 분당 30건의 URL이 시스템에 도착하였다. 이때 시스템 부하를 계산하고, 정상상태에서 시스템 내에 동시에 4개 이상의 URL이 분석 중일 확률을 구하라. 또한, 시스템 부하가 60으로 증가한다면, 5개 이상의 URL이 동시에 분석 중일 확률은 어떻게 변화하는지 분석하라.

- 시스템 유형: $M/G/\infty$ 큐 시스템
- 평균 도착률: $\lambda = 30$URL/분
- 평균서비스시간: $\frac{1}{\mu} = 2초 = \frac{1}{30}$분
- 시스템 부하: $\gamma = \frac{\lambda}{\mu} = 30 \times \frac{1}{30} = 1$
- 동시에 4개 이상의 URL이 분석 중일 확률은 $P(N \geq 4)$를 구하면 됨
  - $P(N \geq 4) = 1 - P(N \leq 3) = 1 - \sum_{n=0}^{3} \frac{1^n \cdot e^{-1}}{n!} \approx 1 - 0.9802 = 0.0198$
- 또한, 시스템 부하가 기존 1에서 60으로 증가하는 경우에 5개 이상의 URL이 동시 분석 중일 확률은 다음 공식으로 계산됨
  - $P(N \geq 5) = 1 - P(N \leq 4) = 1 - \sum_{n=0}^{4} \frac{60^n \cdot e^{-60}}{n!}$



(그림 10) 시스템 부하 증가에 따라 5개 이상의 URL이 동시 분석 중일 확률 그래프

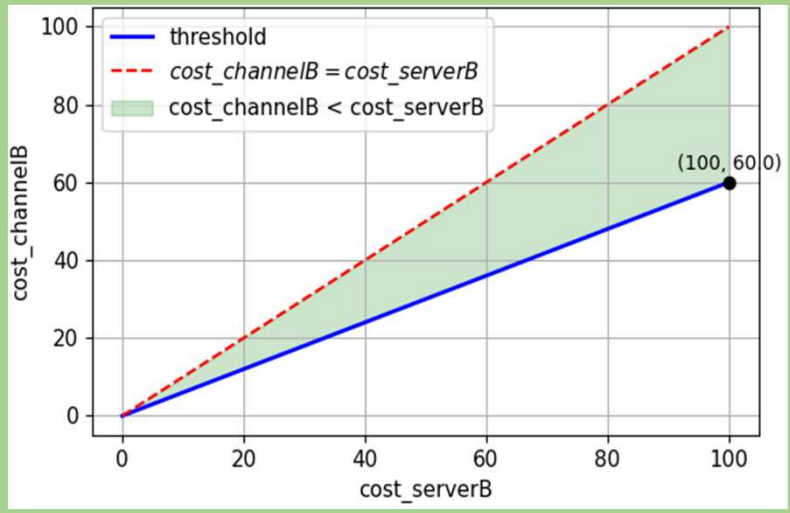
∴ 동시에 4개 이상 분석 중일 확률은 약 1.98%이고,
시스템 부하가 증가함에 따라 5개 이상 동시 분석 중일 확률은 증가하는 것을 볼 수 있음

*추가예제 3 응용 문제

추가예제 10

한 통합 네트워크 서버는 추가예제 3과 같은 조건을 가진다. 기존 기법에서는 서버 1개 증설 비용이 $cost_{serverA} = 10$, 그에 따른 k개의 채널 유지비를 $cost_{channelA} = \frac{cost_{serverA}}{5} \times k$, 즉 $cost_{channelA} = 2k$라고 하자. 제안 기법에서는 서버 1개 증설 비용이 $cost_{serverB}$, 그에 따른 k개의 채널 유지비를 $cost_{channelB} = \frac{cost_{serverB}}{5} \times k$라고 하자. 이때 채널 점유 수 $k = 3$이라고 고정된다고 하자. 이때 제안 기법의 $cost_{channelB}$가 얼마 이하여야 $cost_{serverB}$보다는 작으면서 기존 기법에서의 브러킹확률과 제안 기법의 브러킹확률은 유사한지 분석하라.

- 제안 기법의 $cost_{channelB} < cost_{serverB}$인 경우를 구하는 문제로, $cost_{channelB} = \frac{cost_{serverB}}{5} \times k$에 대입해보면 다음과 같음
- $\frac{cos\ _{serverB}}{5} \times k < cost_{serverB}$, 즉 $k < 5$이면 임계값을 구할 수 있게 됨
- 파란선: $cost_{channelB} = \frac{cost_{serverB}}{5} \times 3$ ($k = 3$ 기준 임계선)
- 빨간선: $cost_{channelB} = cost_{serverB}$ (cost가 동일해지는 기준)
- 검은점: 임계치 예시 중 하나로, $cost_{serverB} = 100$일 때 $cost_{channelB} = 60$ 이하여야 함을 의미함
- 녹색 부분: 제안 기법이 기존 기법과 브러킹확률이 동일한 상황에서, cost 측면에서의 우위를 가지는 영역에 해당함



(그림 11) 제안 기법의 $cost_{channelB}$ 임계값 분석 그래프

감 사 합 니 다

김 지 혜 (jihye@pel.sejong.ac.kr)

V 부록 #1 - 주요용어

- 상태 전이: State Transition
- 연속시간 마코프 과정: Continuous-Time Markov Process
- 평균도착률: Average Arrival Rate
- 평균서비스율: Average Service Rate
- 단위 시간: Unit Time
- 평균도착간격: Average Interarrival Time
- 평균서비스시간: Average Service Time
- 생성률: Birth Rate
- 소멸률: Death Rate
- 상태확률: State Probability
- 상태전이방정식: State Transition Equation
- 평행상태: Steady State
- 평균 트래픽 강도: Average Traffic Intensity
- 도착과정: Arrival Process
- 서비스과정: Service Process
- 트래픽 부하: Offered Load
- 시스템 폭주 확률: Blocking Probability
- 유효 도착률: Effective Arrival Rate
- 체재 시간: System Sojourn Time
- 평균 체재 시간: Mean Sojourn Time
- 평균 대기 시간: Mean Waiting Time
- 대기 시간 분포: Waiting Time Distribution
- 시스템 내 평균 고객 수: Average Number of Customers in System
- 큐 내 평균 대기 고객 수: Average Number of Waiting Customers in Queue

V 부록 #1 - 주요용어

- 일괄 크기(한 번에 도착하는 고객 수): Batch Size
- 일괄 작업(동시에 처리되는 집단 요청): Batch Job
- 집단 도착(여러 요청이 동시에 도착): Batch Arrival
- 재귀식(상태 확률 계산을 위한 반복 수식): Recursive Formula
- 일량 보존 시스템(자원이 쉬지 않고 일함): Work-Conserving System

V 부록 #2 - 복수서버 대기행렬 시스템 종류

- $M/M/c$: 단일 도착/서비스, 다수 서버, 무제한 큐
- $M/M/c/K$: 제한된 큐 크기 포함 (총 수용량 K)
- $M/M/c-B$: 고객이 진입 전 판단함 (Balking 포함)
- $M/M/c-R$: 고객이 기다리다 도중 이탈 (Reneging 포함)
- $M^X/M/c/c$: Batch 도착 + 다수 서버 + no queue
- $M/D/N@c$: 주기적 deterministic 서비스 + bulk 처리
- $M/G/\infty$: 무한 서버 + 일반 분포 서비스 시간
- $M/X/\infty$: 서비스 분포가 미정인 무한 서버 시스템

V 부록 #3 - Python 코드

- (그림 3)

```
1.  import matplotlib.pyplot as plt
2.  import numpy as np
3.  plt.rcParams.update({'font.size': 11})

4.  c = 3
5.  T = 30
6.  N = 6

7.  time_points = np.arange(0, T + 1, c)
8.  processed_customers = [3, 2, 4, 6, 3, 2, 4, 4, 5, 2, 6]

9.  plt.figure(figsize=(5.5, 4))
10. bars = plt.bar(time_points, processed_customers, width=2.5, align='center', color='skyblue',
    edgecolor='black')

11. for bar in bars:
12.     height = bar.get_height()

13. plt.xticks(np.arange(0, T + 1, c))
14. plt.yticks(range(0, N + 2))
15. plt.xlabel('Time')
16. plt.ylabel('Number of Customers')
17. plt.title(f'M/D/{N}@{c}')
18. plt.grid(axis='y', linestyle='--', alpha=0.7)
19. plt.tight_layout()
20. plt.show()
```

V 부록 #3 - Python 코드

- (그림 5)(1/2)

```
1.  import matplotlib.pyplot as plt
2.  import numpy as np
3.  plt.rcParams.update({'font.size': 11})

4.  arrival_times = [0.5, 1.5, 2.8, 4.2, 6.3]
5.  service_times = [0.5, 1.8, 3.5, 2.4, 1.4]
6.  t = 5

7.  x_line = np.linspace(0, t, 100)
8.  y_line = t - x_line

9.  colors = []
10. for i, (x, y) in enumerate(zip(arrival_times, service_times)):
11.     if y > t - x and x <= t:
12.         colors.append('black')
13.     else:
14.         colors.append('white')

15. plt.figure(figsize=(5.5, 4))
16. for x, y, c in zip(arrival_times, service_times, colors):
17.     plt.scatter(x, y, color=c, edgecolor='black', s=100)

18. plt.plot(x_line, y_line, color='black', label=r'$y = t - x$')
19. plt.axvline(x=t, linestyle='--', color='gray', label=r'$x = t$')
```

- (그림 5)(2/2)

```
20. plt.xticks(arrival_times, [f"$A_{i+1}$" for i in range(len(arrival_times))])
21. plt.xlabel("Arrival Time")
22. plt.ylabel("Service Time")
23. plt.title("Marked Poisson Process")

24. plt.xlim(0, 7)
25. plt.ylim(0, 6)
26. plt.grid(True)
27. plt.legend()
28. plt.tight_layout()
29. plt.show()
```

V 부록 #3 - Python 코드

- (그림 6)(1/2)

```
1.  import numpy as np
2.  import matplotlib.pyplot as plt
3.  plt.rcParams.update({'font.size': 11})

4.  c = 4
5.  mu = 2
6.  lambda_1 = 1
7.  p3 = 0.3
8.  p4 = 0.2
9.  lambda_2_range = np.linspace(0, 2, 100)

10. def calculate_utilization(lambda_1, lambda_2_range, p3, p4, c, mu):
11.     utilization = []
12.     for lambda_2 in lambda_2_range:
13.         term1 = 1 * lambda_1 * (1 - p4)
14.         term2 = 2 * lambda_2 * (1 - (p3 + p4))
15.         psi = (term1 + term2) / (c * mu)
16.         utilization.append(psi)
17.     return utilization

18. utilization_values = calculate_utilization(lambda_1, lambda_2_range, p3, p4, c, mu)
19. plt.figure(figsize=(5.5, 4))
20. plt.plot(lambda_2_range, utilization_values, label='System Utilization Ψ', color='blue')
21. plt.axhline(y=1.0, color='red', linestyle='--', label='Max Utilization (100%)')
22. plt.xlabel('λ₂')
```

- (그림 6)(2/2)

```
23. plt.ylabel('System Utilization(₩)')
24. plt.grid(True)
25. plt.legend()
26. plt.tight_layout()
27. plt.show()
```

V 부록 #3 - Python 코드

- (그림 7)

```
1.  import numpy as np
2.  import matplotlib.pyplot as plt
3.  from scipy.stats import poisson
4.  plt.rcParams.update({'font.size': 11})

5.  c_values = [5, 6, 7, 8, 9]
6.  lambda_vals = np.linspace(0.1, 1.5, 100)

7.  drop_threshold = 4
8.  plt.figure(figsize=(6, 4))

9.  for c in c_values:
10.     drop_probs = []
11.     for lam in lambda_vals:
12.         gamma = lam * c
13.         drop_prob = 1 - poisson.cdf(drop_threshold - 1, mu=gamma)
14.         drop_probs.append(drop_prob)
15.     plt.plot(lambda_vals, drop_probs, label=f'c = {c}')

16. plt.xlabel('Arrival Rate (λ)')
17. plt.ylabel('Drop Probability $P(k \\geq 4)$')
18. plt.grid(True)
19. plt.legend()
20. plt.tight_layout()
21. plt.show()
```

V 부록 #3 - Python 코드

- (그림 8)(1/2)

```
1.  import numpy as np
2.  import matplotlib.pyplot as plt
3.  from scipy.stats import poisson
4.  plt.rcParams.update({'font.size': 11})

5.  N = 2
6.  c_list = [10, 5]
7.  lambda_values = np.linspace(0.01, 1.0, 100)

8.  def compute_avg_queue_length(lam, c, N):
9.      gamma = lam * c
10.     L = 0
11.     for k in range(0, 100):
12.         pk = poisson.pmf(k, mu=gamma)
13.         if k > N:
14.             L += (k - N) * pk
15.         if pk < 1e-10:
16.             break
17.     return L

18. plt.figure(figsize=(9, 3.5))
19. for c in c_list:
20.     L_values = [compute_avg_queue_length(lam, c, N) for lam in lambda_values]
21.     label = f'c = {c}s'
22.     plt.plot(lambda_values, L_values, label=label)
```

V 부록 #3 - Python 코드

- (그림 8)(2/2)

```
23. plt.xlabel('Arrival Rate (λ)')
24. plt.ylabel('Average Number of Customers (L)')
25. plt.grid(True)
26. plt.legend()
27. plt.tight_layout()
28. plt.show()
```

V 부록 #3 - Python 코드

- (그림 9)

```
1.  import numpy as np
2.  import matplotlib.pyplot as plt
3.  from scipy.stats import poisson
4.  plt.rcParams.update({'font.size': 11})

5.  gamma = 2

6.  k_vals = np.arange(0, 15)
7.  cdf_vals = poisson.cdf(k_vals, mu=gamma)
8.
    plt.figure(figsize=(5.5, 4))
9.  plt.plot(k_vals, cdf_vals, marker='o', linestyle='-', label='P(N ≤ k)')
10. plt.axhline(0.99, color='r', linestyle='--', label='y=0.99')
11. plt.axvline(6, color='g', linestyle='--', label='k = 6')

12. plt.xlabel('k')
13. plt.ylabel('Probability')
14. plt.grid(True)
15. plt.legend()
16. plt.tight_layout()
17. plt.show()
```

V 부록 #3 - Python 코드

- (그림 10)

```
1.  import numpy as np
2.  import matplotlib.pyplot as plt
3.  from scipy.stats import poisson
4.  plt.rcParams.update({'font.size': 11})

5.  gamma_values = np.arange(1, 61)

6.  k_threshold = 5
7.  prob_ge_5 = [1 - poisson.cdf(k_threshold - 1, mu=gamma) for gamma in gamma_values]

8.  plt.figure(figsize=(5.5, 4))
9.  plt.plot(gamma_values, prob_ge_5, marker='o', color='orange', linestyle='-', label='P(N >= 5)')
10. plt.xlabel('$\gamma$')
11. plt.ylabel('Probability')
12. plt.grid(True)
13. plt.legend()
14. plt.ylim(0, 1.05)
15. plt.tight_layout()
16. plt.show()
```

V 부록 #3 - Python 코드

- (그림 11)

```
1.  import matplotlib.pyplot as plt
2.  import numpy as np
3.  plt.rcParams.update({'font.size': 11})
4.  k = 3
5.  divisor = 5
6.  cost_serverB_values = np.linspace(0, 100, 500)
7.  cost_channelB_values = (cost_serverB_values / divisor) * k
8.  identity_line = cost_serverB_values
9.  plt.figure(figsize=(6, 4))
10. plt.plot(cost_serverB_values, cost_channelB_values, label=fr"threshold", color='blue', linewidth=2)
11. plt.plot(cost_serverB_values, identity_line, label=r"$cost\_channelB = cost\_serverB$", color='red',
    linestyle='--')
12. plt.fill_between(cost_serverB_values, cost_channelB_values, identity_line, where=cost_channelB_values
    < identity_line, color='green', alpha=0.2, label="cost_channelB < cost_serverB")
13. threshold_x = 100
14. threshold_y = (threshold_x / divisor) * k
15. plt.scatter([threshold_x], [threshold_y], color='black', zorder=5)
16. plt.text(threshold_x, threshold_y + 2, f"({threshold_x}, {threshold_y:.1f})", fontsize=10, ha='center',
    va='bottom')
17. plt.title("cost_channelB vs cost_serverB (k=3)")
18. plt.xlabel("cost_serverB")
19. plt.ylabel("cost_channelB")
20. plt.grid(True)
21. plt.legend()
22. plt.tight_layout()
23. plt.show()
```