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

보충

I

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

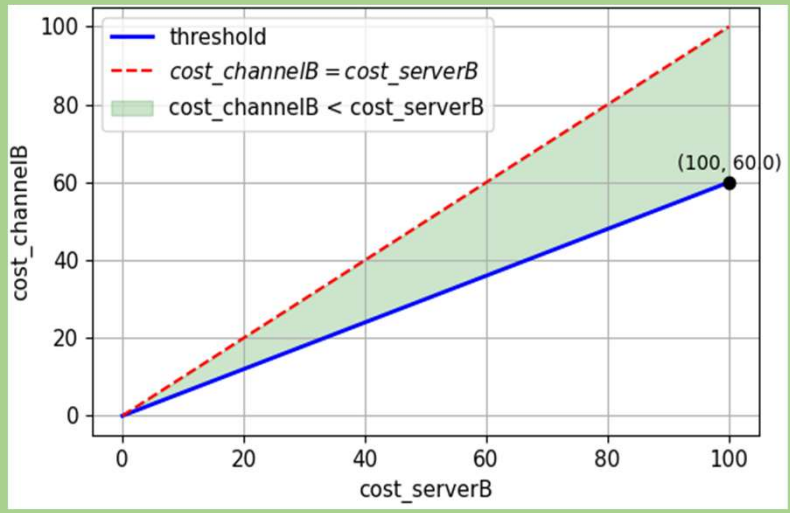
IV 4 추가예제

*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
- $\frac{cost_{serverB}}{5} \times k < cost_{serverB}$, 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

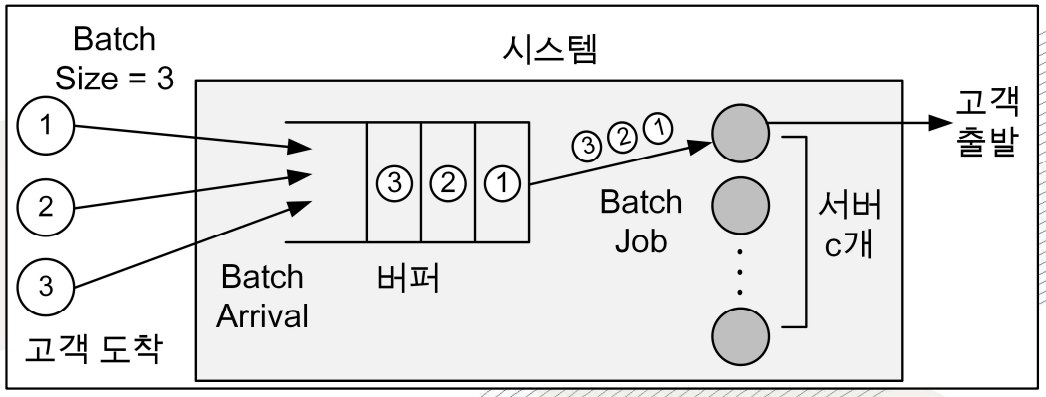
Batch Job을 가진 $M^X/M/c/c$ 큐

II

II 2 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

II 2 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

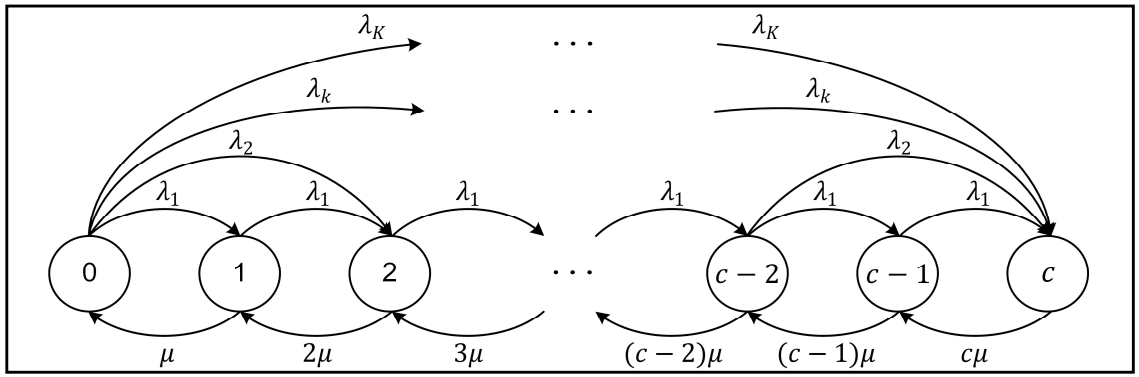
C. $M^X/M/c/c$ 큐의 상태확률

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

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

증명#1

상태전이방정식 경우#1 활용

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

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

감 사 합 니 다

김 지 혜 (jihye@pel.sejong.ac.kr)

III 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

III 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

III 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

III 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()
```