한국시뮬레이션학회 논문지
Vol. 16, No. 2, pp. 75-81 (2007. 6)

도착 및 이탈시점에 근거한 관측 불가능한 후입선출 대기행렬 모형의 분석

김윤배[1] · 박진수[1†]

Analysis of Unobservable Queueing Model with Arrival and Departure Points: LCFS

Yun Bae Kim · Jinsoo Park

ABSTRACT

Previous queue inference has been studied with some limits. Larson's inference engine, which is the basis for this paper, also processed with basic assumption that arrival process is poisson process. Our inference method, which relaxes the poisson process assumption, must be a useful tool for looking into unobservable inside of queueing systems, as well as calculating accurate system performance. This paper employs these inference methods and proves the validity. Then we apply this method to system analysis for more complicated models. At first, we suggest methods to system with known number of servers, then expand to unknown number of servers. For validating our inference approach, we run some simulation models and compare true values with our results.

Key words : *Queue inference, Optimization, Simulation*

요 약

과거의 대기행렬에 대한 추론은 여러 가지 제약 조건이 포함되어 수행되었다. 본 논문의 기반이 되는 Larson의 추론엔진 또한 포아송 도착과정을 기본 가정으로 추론을 수행하였다. 그러나, 이러한 가정 없이 실측 데이터만을 가지고 추론을 수행할 수 있다. 이는 보다 정확한 시스템의 성능척도 계산은 물론 보이지 않는 시스템 내부를 규명하는데 유용한 도구가 될 것이다. 본 논문은 이러한 추론 방법을 제안하고 타당성을 검토하여 보다 나은 시스템 분석에 적용하고자 한다. 먼저 서버의 수를 알고 있는 경우에 대한 추론 방법을 소개하고, 이를 바탕으로 서버의 수를 모르는 경우의 추론에 대해 확장한다. 우리가 제안한 추론모델을 검증하기 위하여 시뮬레이션을 수행하고 실제 시스템 성능척도 값과 추론에 의한 값을 비교 검토하였다.

주요어 : 대기행렬 추론, 최적화, 시뮬레이션

1. 서 론

대기행렬시스템의 추론은 Larson(1990) 이 서비스 시작시점과 종료시점을 알고 있다는 가정 하에 대기열의 행태를 추론하였다[11]. 그는 포아송 도착과정을 기본 가정으로 하여 대기열을 추론하는데, 이 방법은 먼저 포아송 도착과정을 가정하고 N명의 고객 도착을 순서통계량으로 놓으면, 이들의 도착간격 분포가 균일분포를 따른다는 것에 착안하여 수학적인 추론 값을 구한다. 이후, 그의 추론 엔진을 기반으로 계산 시간을 단축시키며 가정의 일반화를 꾀하는 추가적인 논문이 여러 편 발표되었다. 또한 이와 비슷한 방향으로 접근한 다른 추론방법들이 나오기도 했다. 이러한 추론은 고객의 도착이 포아송이라는 가정이 무너지면 전혀 사용할 수 없으며, 시스템의 정확한 행태를 관찰하기도 어렵다. 본 연구의 선행연구로 우리는 이들의 단순한 추론방법이 아닌, 선입선출을 가정한 대기행렬 시스템에서의 입력시점과 이탈시점만을 이용해 정확한 대기행렬의 내부 동작을 추론할 수 있도록 시도하였다[14]. 본 논문에서는 이를 후입선출인 경우에 확장하여 추론을 시도하였다. 물론 선행연구와 마찬가지로 입력시점과 이탈시점만을 가지고 추론을 하므로 안정상태의 가정이나

2007년 5월 17일 접수, 2007년 6월 18일 채택
[1] 성균관대학교 시스템경영공학과
주 저 자 : 김윤배
교신저자 : 박진수
E-mail; kimyb@skku.edu

도착과정, 혹은 서비스과정에 대한 정보가 전혀 없어도 대기행렬시스템에 관한 추론이 가능하다[1].

본 논문의 구성은 2장에서 먼저 선행연구인 선입선출 시스템에 대한 개략적인 추론 방법을 설명한다. 3장에서는 서버의 수를 알고 있는 경우에 대하여 우리가 개발한 추론방법을 먼저 간단히 소개하고 서버의 수를 모르는 경우에 대한 일반적인 추론 모형을 소개한다. 4장에서는 우리가 개발한 모형에 대한 타당성을 보기 위해 실험한 내용과 그 결과를 보여준다. 5장에서는 간단한 요약과 추후에 더 연구 및 보완해야 할 내용을 언급한다.

2. 선행연구의 고찰 (선입선출 시스템 : FCFS)

2.1 서버의 수를 알고 있는 경우

우리가 알고 있는 시스템 성능척도는 총 고객 수, 각 고객들의 도착시점, 이탈시점 뿐이다. 이러한 자료만 있다면 고객들의 도착간격이나 서비스 시간, 심지어는 시스템의 정상상태(stationarity) 가정마저 무시해도 정확한 추론을 할 수 있다.[2] 여기서 우리의 목적은 바쁜 기간 동안의 각 고객의 대기시간과 서비스 시간을 도출해 내는 것이다.

먼저 우리는 추론을 위해 다음과 같은 성능척도들에 대한 용어들을 정의한다. 이 중 실측된 데이터는 A_i, D_i, N 뿐이며 나머지는 모두 이들로부터 계산된 값이거나 추론에 의해 나오는 결과 값들이다.

N : 시스템 내 서비스 받고 나간 총 고객 수
A_i : i 번째 고객의 도착 시점
D_i : i 번째 고객의 이탈 시점
$D_{(k:m)}$: 처음 m 개의 이탈시점들의 k번째 순서 통계량
B_i : i 번째 고객의 서비스 시작 시점
Q_i : i 번째 고객의 대기시간
S_i : i 번째 고객의 서비스 시간
T_i : i 번째 고객의 시스템 체재시간
$(D_i - A_i = Q_i + S_i)$
c : 서버의 수

서버의 수를 알고 있는 경우의 대기시간 예측은 간단하다. 만일 서버의 수가 c개라고 하면 최초 c명의 고객들은 도착과 동시에 서비스를 받기 시작할 것이다.

$$B_i = A_i, \quad i = 1, \cdots, c \tag{1}$$

c개의 서버가 모두 서비스를 제공하기 시작한 이후에 도착하는 고객들은 유휴한 서버가 존재하면 도착과 동시에 서비스를 받고 그렇지 않은 경우에는 현재 서비스 받고 있는 고객들 중 가장 먼저 이탈하는 고객이 이탈함과 동시에 서비스를 받기 시작할 것이다.

$$B_i = \max\{A_i,\ D_{(i-c:i-1)}\},\ i = c+1, \cdots, N. \tag{2}$$

이로부터 다음과 같이 간단히 그 대기시간과 서비스 시간의 정확한 값을 추정할 수 있다.

$$Q_i = B_i - A_i \tag{3}$$

$$S_i = D_i - B_i \tag{4}$$

이로부터 우리는 여러 가지 성능척도들을 계산해 낼 수 있다. 도착시점과 이탈시점을 알면 시스템으로 도착하는 고객과 이탈하는 고객의 궤적을 그려서 현재 시스템내의 고객수를 계산해 낼 수 있다. 또한 서비스 시작점에 대해서도 현재 시스템내의 대기 고객수를 계산해 낼 수 있으며, 서버의 바쁠 확률도 계산해 낼 수 있음은 물론 평균대기고객수나 평균대기시간, 평균서비스시간 등의 전체 성능척도의 계산도 당연히 가능하다.

2.2 서버의 수를 모르는 경우

이제 서버의 수를 모르는 경우에 대해 논하여보자. 이는 대기행렬 시스템의 내부 동작을 전혀 알 수 없는 상황이다. 여기서는 식 (2)와 같은 대기열의 형태가 바로 보이지 않게 된다. 이는 서버의 수인 c 의 값이 미지수이기 때문이다. 결국 c 와 B_i 들로 이루어진 방정식의 형태를 띠게 된다.

이와 같은 방정식을 풀기 위해 다음과 같은 시도를 해보자. 먼저 적당한 c값을 정한 후, 식 (1)과 (2)에서의 방법으로 대기행렬 시스템의 대기시간을 추론한다. 이러한 경우에 c 는 정확한 추론 값이 아니라 단순한 추정치일 뿐이다. 이를 보완하기 위해 최적의 c 와 B_i 들을 찾아낼 수 있는 최적화 식이 필요하다. 최적화 식의 제약조건은 위의 서버의 수를 알고 있는 경우라 가정한 상태에서 서비스 시간을 추론하여 이를 가시영역에 포함되도록(≥ 0) 하면 된다. 목적함수는 정확한 c 의 값을 찾아 줄 수 있는 형태를 구해야 하는데 결과적으로 서비스 시간의 추정치

1) 특히 각 서버의 서비스 시간에 대한 분포가 동일하지 않더라도 추론이 가능하다.
2) 시뮬레이션을 통한 추론에서는 시스템이 정상상태에 도달한 시점 직후의 자료만을 수집해야 하지만, 본 연구에서는 이 가정을 완화 시킬 수 있어 시스템의 상태에 상관없이 정확한 추론을 수행할 수 있다.

인 $\hat{S}$의 분산이 최소화되는 서버의 수를 정확한 c 의 값으로 추론할 수 있다. 이는 서버의 수를 원래 시스템의 서버 수 보다 작거나 크게 잘못 추정하는 경우, 추론한 서비스 시간 $\hat{S}$ 의 값이 실제 값 S 보다 작거나 크게 되어 즉, $\hat{S}=S+\nabla$ 꼴이 되어 분산 값이 커짐을 짐작할 수 있기 때문이다. 서버의 수를 가정하고나면 $\hat{S}_i$ 들은 $\hat{D}_i$ 와 $\hat{B}_i$ 의 차이를 이용해 계산해 낼 수 있다. 즉, $\hat{S}_i=\hat{D}_i-\hat{B}_i$ 이다. 그러므로 다음과 같은 최적화 문제의 해를 얻음으로써 우리가 원하는 대기시간에 대한 정확한 추론 값을 얻어낼 수 있게 된다.

$$\underset{\hat{c}}{Minimize} \quad \sum_{i=1}^{N}\frac{\left(\bar{S}-\hat{S}_i\right)^2}{N-1} \tag{5}$$

$$s.t.\ \hat{B}_i=A_i, \quad i=1,\cdots,\hat{c} \tag{6}$$

$$\hat{B}_i=\max\left\{A_i,\ D_{(i-\hat{c},\,i-1)}\right\},\ i=\hat{c}+1,\cdots,N \tag{7}$$

$$\hat{S}_i=D_i-\hat{B}_i \tag{8}$$

$$\hat{S}_i>0 \tag{9}$$

이 문제를 풀기 위해서는 먼저 $\hat{c}$ 를 결정한 후 $\hat{B}_i$ 를 구하고 이로부터 $\hat{S}_i$ 를 구하여 그 분산을 계산한다. $\hat{c}$ 의 값을 바꾸어가며 $\hat{S}_i$ 의 분산이 최소가 되는 $\hat{c}$ 를 구하게 되면 원래 시스템을 정확히 추론할 수 있다. 결국 이 최적화 문제를 이용하여 완전한 서버의 수와 대기시간을 얻어낼 수 있다. 대기열의 행태나 성능척도를 계산하는 방법은 앞 절에서 사용한 식을 그대로 이용하면 될 것이다. 그러나 여전히 한 가지 문제점이 존재한다. 바로 예외적인 해가 발생한다는 점이다. 고객의 도착간격과 서비스 시간이 확정적인 분포, 즉 상수 값일 경우, 혹은 서버의 이용률 ρ 가 매우 작은 경우에 대해서는 무한한 해가 존재하게 된다. 이는 시스템 내의 고객수가 서버의 수 c 보다 항상 작기 때문이다. 이러한 경우에 대해서는 식 (5)를 만족하는 최소의 c 값을 채택하기를 권장한다. 하지만 실제로 서버의 이용률이 아주 작은 이러한 시스템은 분석에 있어 그 가치가 상대적으로 적으므로 무시할 수 있다.

3. 후입선출 시스템 : LCFS

3.1 서버의 수를 알고 있는 경우

우리가 알고 있는 시스템 정보는 서비스를 받고 나간 총 고객 수, 도착시점, 이탈시점 그리고 서버의 수이다. 시스템은 후입선출(LCFS) 정책으로 서비스를 한다. 다음과 같이 이미 알고 있는 시스템 정보에 대한 용어를 정의한다.

N : 시스템 내 서비스 받고 나간 총 고객 수
A_i : i 번째 고객의 도착 시점
D_i : i 번째 고객의 이탈 시점
c : 서버의 수

고객의 도착 시점과 이탈 시점을 알고 있다면 이로부터 각 고객의 도착과 이탈 시점에서의 시스템 고객수를 구해낼 수 있다. 또한 이로부터 다음을 정의하고 구해낼 수 있다.

N_i^A : i 번째 고객이 도착하면서 보는 고객 수
N_i^D : i 번째 고객이 이탈하면서 보는 고객 수
$D^1(n,t)$: 시점 t 이후 최초 $N_i^D=n$인 이탈시점

다음과 같은 식을 이용하여 서비스 시작점을 구할 수 있다.

$$B_i=\begin{cases} A_i & (N_i^A<c) \\ D^1(N_i^A,\ A_i) & (o/w) \end{cases} \tag{10}$$

시스템에 도착하면서 자신을 제외한 고객수가 c 보다 작다면 모두 서비스 중이고 서버가 하나이상 유휴한 경우이므로 도착하자마자 서비스를 시작한다. 그렇지 않은 경우는 모든 서버가 서비스를 하고 있는 중이 된다. 이 경우 LCFS에서는 자신 이후에 도착한 모든 고객을 서비스 한 후 자신의 서비스가 시작되게 된다. 따라서 자신이 들어오면서 본 고객수와 동일해지는 순간 자신의 서비스를 시작하게 된다. 좀 더 자세히 설명하기 위해 그림 1을 보자. 고객이 들어오면서 자신을 제외하고 n 명을 바라본다면 현재 c 명은 서비스 중이고 $n-c$ 명은 대기 중이다. LCFS는 특성상 자신이 대기열의 제일 앞으로 간다. 이후의 과

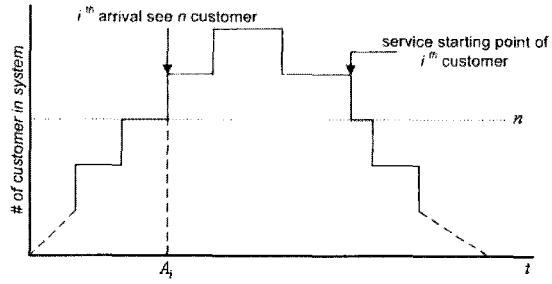


그림 1. LCFS 시스템에서의 서비스 시작점

정이 어찌 되었던 간에 현재 서비스 받는 고객들과 자신보다 늦게 도착한 고객들이 모두 서버를 차지하고 나야지만 자신의 차례가 된다. 그 때가 바로 n 명을 남기고 이탈하는 고객이 있을 때이다. 이탈이 일어났다는 것은 유휴한 서버가 생겼다는 뜻이고 n 명이 남았다는 것은 서비스 받는 고객 수 $c-1$ 명(이탈이 일어났으므로 서버 하나는 유휴한 상태이다)과 자신을 포함한 $n-c+1$ 명은 대기열에 서 있다는 뜻이다. 대기열에 자신을 포함하여 $n-c+1$ 명이 대기 중이라면 이는 분명 그 고객과 그 고객 이전에 온 고객들로만 이루어진 대기열이다. 따라서 현재 유휴한 서버가 하나 생겨났으므로 대기하는 고객 중 가장 늦게 도착한 자신의 서비스 시간이 되는 것이다.

3.2 서버의 수를 모르는 경우

서버의 수를 모르는 경우에 있어 우리가 가장 먼저 떠오른 것은 선행연구에서와 마찬가지로 추정된 서비스시간의 분산을 최소화하는 방법이다. 놀랍게도 이 방법은 후입선출에도 그대로 적용 가능하였으며, 제약 함수를 다음 식 (12), (13), (14)와 같이 이용하면 된다. 그 타당성에 대한 결과가 4장의 실험결과로 증명 되어있다.

$$\underset{\hat{c}}{Minimize} \quad \sum_{i=1}^{N} \frac{\left(\bar{S}-\hat{S}_i\right)^2}{N-1} \tag{11}$$

$$s.t.\ \hat{B}_i = \begin{cases} A_i & (N_i^A < \hat{c}) \\ D^1(N_i^A, A_i) & (o/w) \end{cases} \tag{12}$$

$$\hat{S}_i = D_i - \hat{B}_i \tag{13}$$

$$\hat{S}_i > 0 \tag{14}$$

물론 이로부터 얻은 최적의 $\hat{c}$ 와 각각의 $\hat{B}_i$ 을 이용하여 정확한 서비스시작점과 대기시간을 계산해 낼 수 있다. 이에 대한 결과는 4장에서 이용한 시뮬레이션 결과와 정확하게 맞아 떨어지므로 기록을 생략하였다.

4. 시뮬레이션 수행 및 실험결과

우리는 모델의 타당성을 검증하기위하여 다음의 절차로 실험을 수행하였다. 먼저 시뮬레이션 모델을 만들어 이를 수행한다. 여기서 각 고객의 도착시점과 이탈시점을 얻어내고 추론의 참값으로 쓰기위해 서비스 시작점을 얻어낸다. 다음으로 얻어낸 도착시점과 이탈시점만을 가지고 서버의 수와 각 고객의 서비스 시작점을 추론해 낸다. 실험의 결론부터 말하면 우리의 추론 모델은 아주 정확한 서버 수와 각각의 서비스 시작점을 추론해 낼 수 있다. 물론 실험에 사용된 서버의 수를 이용하여 서버의 수를 알고 있다고 가정하고 추론을 한 값은 참값과 한 치의 오차도 없이 맞아 떨어지는 것을 볼 수 있었으므로 실험 결과에서 생략하였다.

4.1 가상 시뮬레이션 모델

실험에 사용한 시스템은 3가지 도착과정과 3가지 서비스분포의 조합으로 총 9가지 시스템이다. 도착과정으로는 포아송 과정, 도착간격이 확정적인 경우와 수정된 정규분포를 따르는 경우의 3가지이며, 서비스 분포도 마찬가지로 지수분포, 확정적 분포, 수정된 정규분포의 3가지이다. 이 각각의 시스템에 서버의 수를 1개, 3개, 7개일 때에 대해 각각 수행하였으며, 이 총 27가지 시스템에 대해 서버의 서비스율(μ)에 대한 입력율(λ)의 비율 $\rho=\lambda/\mu$을 각각 0.9, 1.0, 1.1의 3가지 형태 시스템으로 나누어 총 81가지 시스템에 대해 시뮬레이션을 수행하였다. 수정된 정규분포로는 평균 $1/\mu$, 분산 1의 정규난수에 절대 값을 취하여 사용하였다. 표 1은 실험에 쓰인 평균값이다.

표 1. 분포에 사용한 모수

평균 도착율(λ) = 1

c	Mean Service Time($1/\mu$)		
	$\rho=0.9$	$\rho=1$	$\rho=1.1$
1	0.9	1	1.1
3	2.7	3	3.3
7	6.3	7	7.7

이와 같은 시스템으로부터 각 고객의 도착시점과 이탈시점, 그리고 서비스 시작점을 각각 1000개씩 추출하였다. 이중 도착시점과 이탈시점을 추론을 위한 데이터로 이용하고 서비스 시작점을 추론에 대한 참값으로 이용하여 추론의 정확성을 검토하였다.

4.2 추론 결과 및 분석

최적화 문제를 푼 결과는 표 2와 같다. 여기서 M은 지수분포, 마코프(Markov, 혹은 포아송) 프로세스, 또는 망각성(memoryless)을 의미하며, GI와 G는 정규분포 $N(1/\mu, 1)$를 따르는 확률변수의 절대 값을 의미한다. 또한 D는 확정적(deterministic) 분포를 말한다. 결과표를 보면서 주의해야 할 점은 본 파라미터를 전혀 모른다는 가정 하에 추론된 결과라는 점이다. 즉, 목적함수 값은 분포의 모수에 따라 주어진 참값(분산)에 근사해야만 한다. 예를 들어 $M/M/3$, $\rho=0.9$의 목적 함수의 경우 참값 $2.7^2=7.29$

표 2. 실험 결과 요약

(소수점 둘째자리 반올림)

A	S	c	$\rho=0.9$		$\rho=1$		$\rho=1.1$	
			$\hat{c}$	obj.	$\hat{c}$	obj.	$\hat{c}$	obj.
M	M	1	1	0.8	1	1.0	1	1.2
		3	3	6.9	3	9.1	3	11.0
		7	7	37.4	7	49.3	7	59.9
	D	1	1	0	1	0	1	0
		3	3	0	3	0	3	0
		7	7	0	7	0	7	0
	G	1	1	0.6	1	0.7	1	0.7
		3	3	1.0	3	1.1	3	0.9
		7	7	1.0	7	1.1	7	0.9
D	M	1	1	0.8	1	1.0	1	1.2
		3	3	6.9	3	9.1	3	11.0
		7	7	37.4	7	49.3	7	59.9
	D	1	1...	0	1,..	0	1	0
		3	3,..	0	3,..	0	3	0
		7	7,..	0	7,..	0	7	0
	G	1	1	0.6	1	0.7	1	0.7
		3	3	1.0	3	1.1	3	0.9
		7	7	1.0	7	1.1	7	0.9
GI	M	1	1	0.8	1	1.0	1	1.2
		3	3	6.9	3	9.1	3	11.0
		7	7	37.4	7	49.3	7	59.9
	D	1	1	0	1	0	1	0
		3	3	0	3	0	3	0
		7	7	0	7	0	7	0
	G	1	1	0.6	1	0.7	1	0.7
		3	3	1.0	3	1.1	3	0.9
		7	7	1.0	7	1.1	7	0.9

에 근사해야 정확한 추론결과이다. 하지만, 본 결과는 시스템이 안정 상태에 도달할 만큼 자료가 충분하지 않기 때문에 오차가 발생하는 것은 감수해야 한다. 또 하나의 주의할 점은 서비스 분포와 서버수가 같을 경우(예를 들어 $M/M/3$, $\rho=0.9$ 인 경우와 $GI/M/3$, $\rho=0.9$ 인 경우)에 대한 목적함수 값이 동일하다는 점이다. 이는 시뮬레이션 수행 시 각 모델에 대하여 동일한 초기 값으로 난수를 발생시켰기 때문이다. 동일 서비스 분포의 난수인 시스템들의 경우 각 시스템의 서비스 과정이 서로 동일해진다. 따라서 목적함수인 그 분산이 동일한 것이다.

결과표에서 c는 시뮬레이션에 사용했던 실제 서버의 수이며 $\hat{c}$는 제시한 목적함수를 최소화시켜 얻은 추론 값이다. 이 결과표에서 우리가 주목해서 보아야 할 부분은 $\rho \leq 1$인 $D/D/c$ 시스템의 경우이다. 이 경우에 무한의

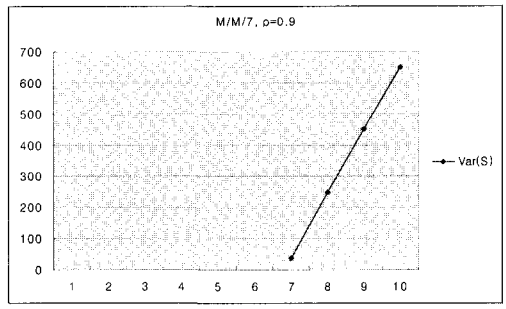


그림 **2.** $M/M/7, \rho=0.9$ 목적함수 값의 변화

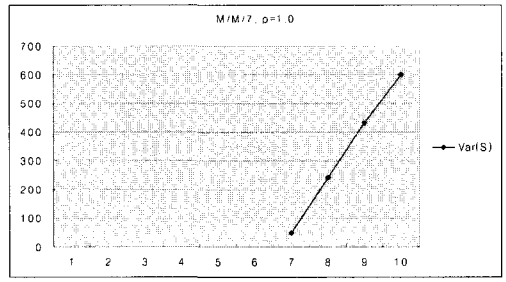


그림 **3.** $M/M/7, \rho=1.0$ 목적함수 값의 변화

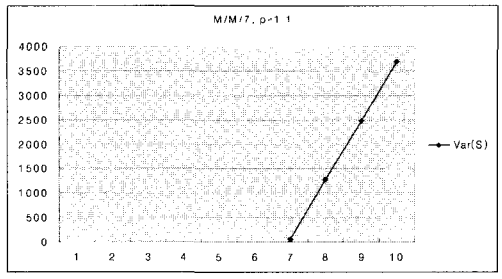


그림 **4.** $M/M/7, \rho=1.1$ 목적함수 값의 변화

해가 존재함을 알 수 있다. 이러한 해가 도출되는 이유는 시스템 내의 고객수가 항상 서버의 수보다 적기 때문이다. 그러므로 도착간격과 서비스 시간의 분포가 불확실성을 동반함에도 불구하고 서버의 이용률이 매우 작으면 역시 무한한 해가 존재할 수 있음을 판단할 수 있다. 그러나 선행연구에서와 마찬가지로 이러한 시스템은 우리의 관심 대상이 아니다. 특히 실제 시스템에서 한명이라도 대기고객이 생기는 경우 ρ의 값에 관계없이 해가 정확히 구해진다. 따라서 본 실험에서 보여주는 결과만으로도 우리의 추론 목표에 적합한 모델임을 설명할 수 있다.

다음으로 목적함수 값의 변화 행태를 살펴보아야 할 필요가 있다. 먼저 그림 2, 3, 4는 $M/M/c$ 형태 시스템 중에서 $M/M/7$ 시스템에 대한 목적함수의 그래프이다.

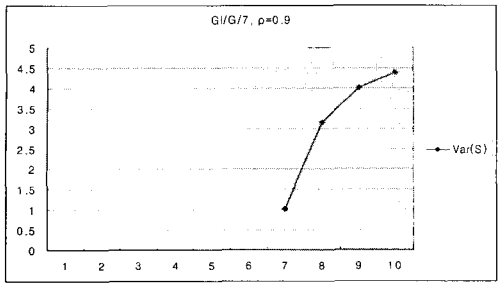


그림 5. $GI/G/7, \rho = 0.9$ 목적함수 값의 변화

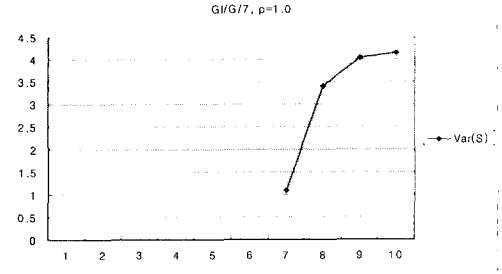


그림 6. $GI/G/7, \rho = 1.0$ 목적함수 값의 변화

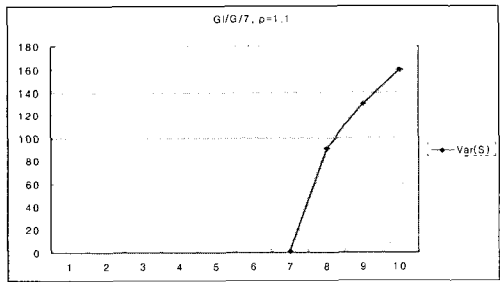


그림 7. $GI/G/7, \rho = 1.1$ 목적함수 값의 변화

이 그림들에서 목적함수 값이 표시되지 않은 부분은 $\hat{S}_i \leq 0$이 되어 제약조건을 위배하는 비가시해(infeasible solution)가 나오는 경우이다.

LCFS 시스템의 경우는 흥미롭게도 대부분의 실험결과에 있어 c의 값이 과소추정되면 비가시해를 갖게 되는 것을 알 수 있다.

다음으로 그림 5, 6, 7은 $GI/G/c$ 형태의 시스템 중에서 $GI/G/7$ 시스템의 목적함수 그래프이다. 역시 표시되지 않은 점은 비가시해를 갖는 결과이다.

이러한 결과 그래프는 우리가 설정한 목적함수인 서비스 시간의 분산이 추론을 위한 최적화 모델의 목적함수로써 타당하다는 것을 입증할 수 있는 결과이다. 모든 그림에서 보듯이 서버의 수가 원래 시스템의 서버수보다 과다추정되면 그때 추정된 서비스 시간의 분산이 크게 증가하는 것을 알 수 있다. 즉 목적함수인 서비스 시간의 분산이 최소가 되는 서버의 수가 본래 시스템의 서버 수와 동일하게 된다는 것이다.

물론 이렇게 추론된 서버의 수로부터 식 (10)을 이용하여 서비스 시작점을 추론해 내면 시뮬레이션 수행에서 얻어낸 서비스 시작점과 완전히 일치함을 볼 수 있다. 이로부터 다양한 시스템 성능척도를 얻을 수 있다는 것은 앞서 언급한 바 있다.

5. 결론 및 추후 연구과제

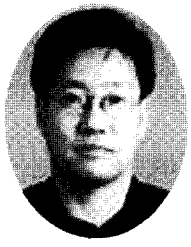
고객의 입력시점과 이탈시점만의 시스템 외적 데이터를 이용하여 대기행렬시스템의 내부 동작을 규명하고자 하는 추론을 시도해 보았다. 이러한 연구들은 Larson의 Queue Inference Engine (QIE)에 근간을 두고 있다. 그러나 QIE 는 서비스 시작점과 서비스 종료시점의 시스템 내부적 요소를 이용하였으며, 고객의 도착과정을 포아송 과정으로 제한하였다. 이에 반해 본 논문에서는 입력시점과 이탈시점의 시스템 외적 요소만으로 내부를 규명하고자 하였다. 본 논문에서 설정한 단 하나의 가정은 각 서버와 서비스 시간이 서로 독립이라는 점이다. 또한 본 연구는 서버의 수를 모르는 경우에도 이를 정확히 추론할 수 있는 방법을 제시하였다.

본 연구에서는 추론을 위해 실험적인 결과를 이용하여 최적화 문제를 모델링하였다. 서비스 시간의 분산이 최소화 된다는 이론적 증명이 포함된다면, 더욱더 확실한 추론방법으로 쓰일 수 있을 것이다. 또한 현재 시스템 가정인 선입선출, 후입선출 뿐 아니라 무작위 추출법에 의한 시스템에 대해서도 추론하고 나아가서는 이러한 서비스 정책을 모를 때에도 추론할 수 있는 모델을 만든다면 일반적인 대기행렬 시스템에 대한 추론의 근간을 마련할 수 있을 것이라 확신한다.

참 고 문 헌

1. Larson, R. (1990), The queue inference engine: Deducing queue statistics from transactional data, *Management Science* 36 586-601.
2. Larson, R. (1991), The queue infe,rence engine: Addendum, *Management Science* 37 (1991) 1062-1062.
3. Bertimas, D. and Servi, L. (1992), Deducing queueing from transactional data: The queue inference engine, revisited, *Operations Research* 40 S217-S228.

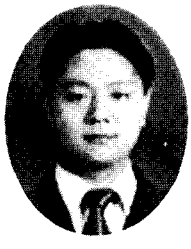
4. Daley, D. and Servi, L. (1992), Exploiting Markov-chains to infer queue length from transactional data, *Journal of Applied Probability* 29 713-732.
5. Dimitrijevic, D. (1996), Inferring most likely queue length from transactional data, *Operations Research Letters* 19 191-199.
6. Mandelbaum, A. and Zeltyn, S. (1998), Estimating characteristics of queueing networks using transactional data, *Queueing Systems* 29 75-127.
7. Jones, L. (1999), Inferring balking behavior from transactional data, *Operations Research* 47 778-784.
8. Masuda, Y. (1995), Exploiting partial information in queueing-systems, *Operations Research* 43 530-538.
9. Basawa, I.,Bhat, U. and Lund, R. (1996), Maximum likelihood estimation for single server queues from waiting time data, *Queueing Systems* 24 155-167.
10. Pickands, J. and Stine, R. (1997), Estimation for the M/G/infinite queue with incomplete information, *Biometrika* 84 295-308.
11. Toyoizumi, H. (1997), Sengupta's invariant relationship and its application to waiting time inference, *Journal of Applied Probability* 34 795-799.
12. Bingham, N. and Pitts, S. (1999), Non-parametric estimation for the M/G/infinity queue, *Annals of the Institute of Statistical Mathematics* 51 71-97.
13. Jang, J., Suh, J. and Liu, C. (1999), A new procedure to estimate waiting time in GI/G/2 systems by server observation, *Computers and Operations Research* 28 (2001) 597-611.
14. Yun Bae, Kim, Jinsoo, Park and Jae-Bum, Kim. Inference for Unobservable Queues from Arrival and Departure Data, *Asian Sim Conference.* (2005)



김 윤 배 (kimyb@skku.edu)

1982년 성균관대학교 산업공학 학사
1986년 University of Florida, Industrial and Systems Engineering 공학석사
1992년 Rensselaer Polytechnic Institute Decision Science and Engineering Systems Ph. D.
1995년~1998년 성균관대학교 산업공학과 조교수
1998년~2004년 성균관대학교 산업공학과 부교수
2004년~현재 성균관대학교 산업공학과 교수

관심분야 : 시뮬레이션 출력분석, 인터넷 트래픽 분석, Telecom Network Performance Analysis



박 진 수 (jsf001@paran.com)

1998년 성균관대학교 산업공학과 학사
2000년 성균관대학교 산업공학과 석사
2000년~현재 성균관대학교 산업공학과 박사과정

관심분야 : 시뮬레이션모델링, 출력분석, Web Performance, 대기행렬이론