Journal of the Korean Institute of Industrial Engineers
Vol.23, NO.4, December, 1997

봉쇄현상이 있는 조립/분해 대기행렬망의 산출율 상한 및 하한에 대한 연구

Throughput Upper and Lower Bounds for Assembly/Disassembly Queueing Networks with Blocking

백천현*

Chun-hyun Paik*

Abstract

Assembly/Disassembly Queueing Networks (ADQNs) with finite buffers have been used as a major tool for evaluating the performances of manufacturing and parallel processing systems. In this study, we present simple but effective methods which yield throughput upper and lower bounds for ADQNs with exponential service times and finite buffers. These methods are based on the monotonicity properties of throughputs with respect to service times and buffer capacities. The throughput-upper bounding method is elaborated on with general network configuration (specifically acyclic configuration). But our lower bounding method is restricted to the ADQNs with more specialized configuration. Computational experiments will be performed to confirm the effectiveness of our throughput-bounding methods.

1. Introduction

The operation of manufacturing and parallel processing systems is often approximately described by a finite-buffered queueing network of assembly/disassembly type. In a manufacturing system, units may be built by assembling multiple subunits and a unit may be disintegrated into one or more subunits for the required operation. In a parallel processing system, disassembling occurs when a job (program) is split into tasks (subprograms) that can be concurrently run on a number of different processors. Assembling occurs whenever a job is allowed to be executed only after the completion of other tasks.

In this paper, we investigate the throughput-bounding methods for ADQNs with finite buffers and exponential service times. Previous studies in the literature has been mostly confined to the ADQNs with infinite buffers [3, 5, 11]. See Liu and Perros [11], Baccelli and Massey [3], and the numerous references therein for the ADQNs with infinite buffers. But only a few studies represent some of the effort that has been dedicated to the analysis of ADQNs with finite buffers [4, 8, 10]. Furthermore, throughput analysis, due to the unavailability of closed-form exact solutions for the ADQNs, has been centered

* 동의대학교 산업공학과

on developing an approximate measure.

As an alternative or a supplement to the approximate approach, one may be interested in bounds which can be obtained with much less computational effort. Within our knowledge, two studies dealing with the throughput bounds for ADQNs with finite buffers have been published [9,10]. Hopp and Simon [9] have obtained bounds for the ADQNs with three-server simple assembly configurations. Lipper and Sengupta [10] have studied assembly networks where all buffers have the same capacity. The objective of this study is to extend to ADQNs with more general configuration.

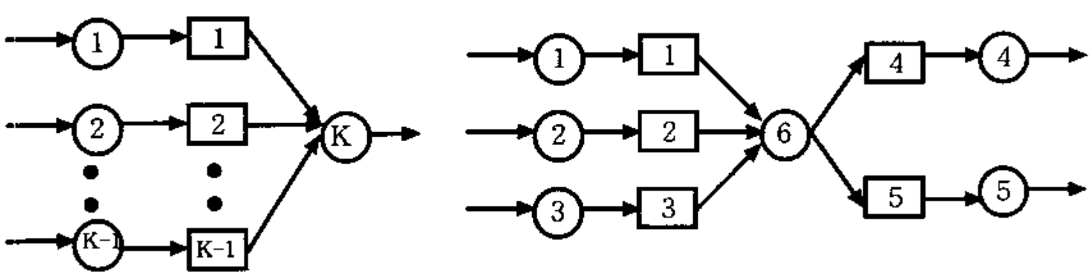
This paper is organized as follows: In Section 2, an ADQN model is described. In Section 3, the throughput-upper and lower bounding methods are presented. In Section 4, extensive computational experiments are conducted, followed by a conclusion in Section 5.

2. Model Description

Consider an ADQN which consists of K service facilities and M finite buffers. Unless otherwise mentioned, we assume there is a single server in each service facility, so service facility i and server i will be used interchangeably with each other. Server i has the sets of upstream and downstream buffers, denoted by $U(i)$ and $D(i)$ respectively. For each buffer j there exist only one upstream and one downstream server, denoted by u_j and d_j, respectively. Assume that the configuration of an ADQN is acyclic (i. e., it does not contain any undirected cycle). Server i with $U(i)=\Phi$, which receives customers from outside the system, is never starved, while server l with $D(l)=\Phi$, which sends customers to outside the system, is never blocked.

A disassembly server splits a departing customer into a number of customers, each to be forwarded one-to-one to its downstream buffers. On the other hand an assembly server may initiate its service only when each of its upstream buffers is occupied, i. e., there is at least one customer waiting to be served at each upstream buffer. A server, just before a service initiation, confirms whether or not its downstream buffers are free. If at least one of these buffers is full, the service initiation is postponed until all downstream buffers are free. This blocking mechanism is called Blocking Before Service (BBS) [12].

To introduce three specialized network configurations, consider the ADQNs with more restricted configuration as follows: server K is the only server to make disassembly and/or assembly operations and $U(u_j)=\Phi$, $j\in U(K)$, $D(d_k)=\Phi$, $k\in D(K)$. This type of ADQNs is specifically called *simple ADQN* (see Fig. 1 (b)). Moreover, networks with $D(K)=\Phi$ and $U(K)=\Phi$ will be called *simple assembly* (see Fig. 1 (a)) and *simple disassembly* networks, respectively.



(a) A simple assembly network (b) A simple ADQNs

Fig 1. Two examples of ADQNs

Service times at server i are distributed according to exponential distribution with rate μ_i, and those of different servers are mutually independent. Let $\{S_{i,n}, n=1,2,...\}$ be the sequence of service times of server i, and denote $\underline{S}=\{S_i, i=1,...,K\}$ where $S_i={}^{st}S_{i,n}$, $n\geq 1$ (note that $X={}^{st}Y$ if $P(X\geq x)=P(Y\geq x)$, $x\in R$ [14]). Customers in each buffer are served in a FIFO (first-in first-out) manner. A server provides its service simultaneously to the customers, one for each of its upstream buffers which is assumed to reside therein while being served. Let $\underline{B}=\{B_1,B_2,\cdots,B_M\}$ where B_j denotes the capacity of buffer j.

The initial condition is given by $\underline{m}=\{m_1,m_2,\cdots,m_M\}$ where m_j represents the number of customers in buffer j at time 0. Let h_j $(=B_j-m_j)$, $j=1,...,M$, be the number of the empty spaces (holes) in buffer j at time 0.

The notation $\Sigma=(\underline{B},\underline{S})$ is used to denote the network with network parameters $(\underline{B},\underline{S})$. Let $\Xi=(\Sigma,\underline{m})$ denote the network coupled with an initial condition $\underline{m}$. The throughputs of Σ and Ξ are denoted by TH(Σ) and TH(Ξ), respectively.

We now establish a set of recursive equations which depicts the evolutionary process of the ADQN, and which will become the basis of the development of our throughput-bounding methods. Let $B_{i,n}$ and $D_{i,n}$ denote the service beginning and departure time of the nth customer at server i, respectively. Assuming that the network Σ starts with the initial condition $\underline{m}$, the following evolution equations represent the system dynamics:

$$B_{i,n}=\max_{j\in U(i),q\in D(i)}\{D_{i,n-1}, D_{u_j,n-m_j}, D_{d_q,n-h_q}\}, \qquad (1)$$

$$D_{i,n}=B_{i,n}+S_{i,n},\ i=1,...,K,\quad n\geq 1, \qquad (2)$$

where $D_{i,n}=0$, $i=1,...,K$ and $n\leq 0$.

It is instructive to note that $B_{i,n}$ is determined by taking into account three conditions: the first is that server i must be available. The second is that all upstream buffers of server i must be non-empty. And finally all downstream buffers of server i must be non-full. The first condition is satisfied when server i has completed its (n−1)th service, which occurs at time $D_{i,n-1}$. The second condition is satisfied when server u_j, $j\in U(i)$, has completed its $(n-m_j)$th service, which occurs at time $D_{u_j,n-m_j}$. The third condition is satisfied when server d_q, $q\in D(i)$, has completed its $(n-h_q)$th service, which occurs at time $D_{d_q,n-h_q}$. Since these conditions must all be satisfied, the instant of beginning of service is the maximum of these three times. Equation (2) simply states that the time of the departure time of the nth customer at server i is equal to the time of its service beginning plus the duration of the service time.

3. Throughput Upper and Lower Bounds

According to our literature survey, there is no study dealing with the approximate methods for finite-buffered ADQNs with general configuration such as ours. In the absence of approximate methods, one may just be interested in conservative but easily obtainable secured bounds in order to get a fast impression of the system performance [9, 10]. Hopp et. al [9] obtained throughput bounds for three-server simple assembly networks and Lipper et. al [10] derived throughput bounds for simple assembly networks where all buffers have the same capacity. In this section, we suggest two throughput-bounding methods with no restriction on the buffer size and the number of servers: one for the upper bounds of ADQNs with acyclic configuration and the other for the lower bounds of simple ADQNs.

3.1 Throughput upper bounds for acyclic ADQNs

Consider an acyclic ADQN, $\Sigma=(\underline{B},\underline{S})$. Let $r(i,l)$ denote the set of servers and buffers on a *route* from server i to server l. Let $r^b(i,l)$ be the subset of $r(i,l)$, composed of only the buffers in $r(i,l)$. Denote the throughput of an ordinary queueing system, M/M/1/B, with arrival rate λ and service rate μ by $f(\lambda,B,\mu)$, i.e.,

$$f(\lambda,B,\mu)=\begin{cases}\lambda\mu(\lambda^B-\mu^B)/(\lambda^{B+1}-\mu^{B+1}) & \text{if } \lambda\neq\mu,\\ \lambda B/(B+1) & \text{if } \lambda=\mu.\end{cases} \quad (3)$$

Lemma 1. Consider two ADQNs, $\Xi^{(k)}=(\Sigma^{(k)},\underline{m}^{(k)})$, $k=1,2$, where $\underline{B}^{(1)}=\underline{B}^{(2)}$, and $\underline{m}^{(1)}=\underline{m}^{(2)}$. Then $S_i^{(1)}\le_{st}S_i^{(2)}$, $i=1,\ldots,K$, implies $D_{i,n}^{(1)}\le_{st}D_{i,n}^{(2)}$.

Proof. The proof is conducted using the *coupling* theorem [16]. Assume that $S_i^{(1)}\sim F_i$ and $S_i^{(2)}\sim G_i$, $i=1,\ldots,K$, and generate successive service times for server i from $S_{i,n}^{(1)}=F_i^{-1}(U_n)$ and $S_{i,n}^{(2)}=G_i^{-1}(U_n)$, where U_n, $n=1,2,\ldots$ represent i.i.d. uniformly distributed random variables. From definition, $S_{i,n}^{(1)}\le S_{i,n}^{(2)}$, almost surely, $n\ge 1$. Hence by the evolution equations (1) and (2), it is evident that $D_{i,n}^{(1)}\le D_{i,n}^{(2)}$, almost surely, $n\ge 1$, $i=1,\ldots,K$. Thus, the proof is completed. □

Lemma 2. Consider two ADQNs, $\Xi^{(k)}=(\Sigma^{(k)},\underline{m}^{(k)})$, $k=1,2$. Then $\underline{S}^{(1)}=^{st}\underline{S}^{(2)}$, $\underline{m}^{(1)}=\underline{m}^{(2)}$, and $\underline{B}^{(1)}\le\underline{B}^{(2)}$ implies $D_{i,n}^{(1)}\ge_{st}D_{i,n}^{(2)}$.

Proof. As shown in equations (1) and (2), the departure times of nth customer at server i, $D_{i,n}$, is determined by three factors : $D_{i,n-1}$, $D_{u_j,n-m_j}$ and $D_{d_q,n-h_q}$. Note that the increase of buffer capacities decreases the value of the third term, $D_{d_q,n-h_q}$ ($h_q=B_q-m_q$), and recursively effects the realization of departure times. From this fact, we deduce that $D_{i,n}$ is decreasing in $\underline{B}$. □

Theorem 1. Consider two acyclic ADQNs, $\Xi^{(k)}=(\Sigma^{(k)},\underline{m}^{(k)})$, $k=1,2$. Then $S_i^{(1)}\ge_{st}S_i^{(2)}$, $i=1,\ldots,K$ or $\underline{B}^{(1)}\le\underline{B}^{(2)}$ implies $\mathrm{TH}(\Xi^{(1)})\le\mathrm{TH}(\Xi^{(2)})$.

Proof. Let $E(X)$ be the expectation of random variable X. By Lemma 1 and 2 and noting the following fact:

$$TH(\Xi)=\lim_{n\to\infty}\frac{n}{E(D_{i,n})},\ i=1,\ldots,K,$$

the proof is immediately completed. □

Remark. Note that Lemma 1, 2 and Theorem 1 hold

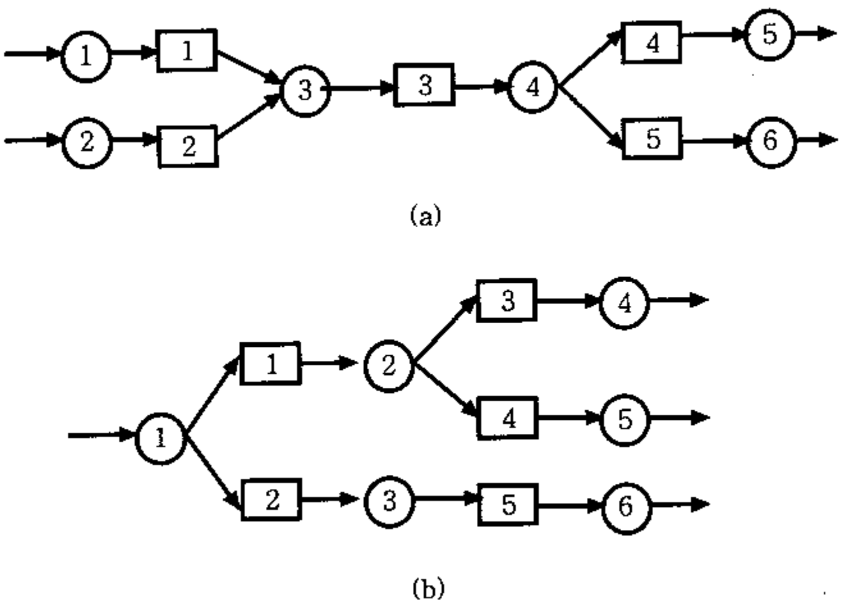


Fig 2. Two six-servers ADQNs

in the general service times, provided that the service times of each server are i.i.d. (independent and identically distributed) random variables.

Since the dynamics of acyclic ADQNs with exponential service times forms an irreducible Markov process, we have the following asymptotic result independent of initial conditions.

Corollary 1. Consider two ADQNs with exponential service times, $\Sigma^{(k)}$, k=1,2. Then $S_i^{(1)} \geq_{st} S_i^{(2)}$, i=1,...,K or $\underline{B}^{(1)} \leq \underline{B}^{(2)}$ implies TH($\Sigma^{(1)}$)≤TH($\Sigma^{(2)}$).

We are now ready to derive throughput upper bounds (TUBs) for acyclic ADQNs with exponential service times. we first construct several decomposed systems, each of which yields a TUB for the original ADQN, as indicated in the following procedure.

Decomposition procedure:

(i) Among all pairs of servers, select the pairs (i,l) such that $r(i,l) \neq \Phi$. Let W be the set of these pairs.

(ii) For a pair $(i,l) \in W$, set $S_k = I$ (that is, $\mu_k = \infty$), $\forall\, k$ $(\neq i,l)$ and $B_j = \infty$, $\forall\, j \in r^b(i,l)$, where I is the unit step function at zero. Let $D(i,l)$ denote the decomposed system constructed by the pair of servers i and l, (i,l).

Note that the decomposed system $D(i,l)$ becomes an isolated M/M/1/B′ system with $B' = \Sigma_{j \in r^b(i,l)} B_j$, arrival rate μ_i and service rate μ_l. For example, consider a decomposed system $D(1,4)$ of the six-server ADQN given in Fig 2 (b). Note that the decomposed system becomes an M/M/1/B with buffer size (B_1+B_3), arrival rate μ_1 and service rate μ_4.

Theorem 2. For an acyclic ADQN with exponential service times, Σ, TH($D(i,l)$), $(i,l) \in W$, is a TUB of Σ. Hence,

$$TH(\Sigma) \leq \min_{(i,l) \in W} \{TH(D(i,l))\} = \min_{(i,l) \in W} \{f(\mu_i, \Sigma_{j \in r^b(i,l)} B_j, \mu_l)\}.$$

Proof. Since $F \geq_{st} I$ for any distribution function F of a positive random variable, TH($D(i,l)$), $(i,l) \in W$, is an upper bound of TH(Σ) by Theorem 1 □

3.2 Throughput lower bounds for simple ADQNs

In this subsection, we derive throughput lower bounds (TLBs) for a simple ADQN Σ. Let $g(\lambda, B, \mu)$ be the empty probability of an isolated queueing system, M/M/1/B, with arrival rate λ and service rate μ, i.e.,

$$g(\lambda, B, \mu) = \begin{cases} (1-\rho)/(1-\rho^{B+1}) & \text{if } \rho \neq 1, \\ 1/(B+1), & \text{if } \rho = 1, \end{cases} \quad (4)$$

where $\rho = \lambda / \mu$.

Lemma 3. The empty probability of buffer j, $j \in U(K)$ in a simple ADQN, Σ, is less than or equal to that of $M/M/1/B_j$ with arrival rate μ_{u_j} and service rate μ_K, i.e., $P_j(N_j=0) \leq g(\mu_{u_j}, B_j, \mu_K)$, $j \in U(K)$.

Proof. By similar arguments as the case of TUB, Hopp et al. [9] showed that the empty probability of buffer j $(j \in U(K))$ in the simple assembly network, in which server K experiences the so-called *starvation delay* (the delay owing to postponing service initiation until each of its upstream buffers is occupied), is less than or equal to that of the associated M/M/1/B. In the simple ADQNs, server K experiences not only starvation delay but also blocking delay. Note that this blocking delay may create some additional opportunity of postponing service initiation and of increasing the number of customers in the upstream buffers of server K. Hence it is obvious that the empty probability of buffer j $(j \in U(K))$ in the simple ADQNs is less than or equal to the one in the simple assembly networks. Thus, the proof is completed. □

Consider another simple ADQN, Σ^r, which is obtained from the original network Σ by exchanging the upstream and downstream servers of each of all buffers. The network Σ^r is called the reverse network of Σ [2, 13]. The notations pertaining to the network Σ^r are denoted by the superscript r.

Lemma 4.([2]) Consider a simple ADQN with exponential service times Σ and its reverse network Σ^r. The blocking probability of a buffer in Σ is equal to the empty probability of the buffer in Σ^r.

Lemma 3 and 4 directly lead to the following theorem.

Theorem 3. For a simple ADQN with exponential service times, Σ,

Table 1. Throughput bounds for K-server simple assembly networks (K=4,5,6)

K	μ_1	μ_2	μ_3	μ_4	μ_5	μ_6	B_1	B_2	B_3	B_4	B_5	Exact	TUB	%err[(1)]	TLB	%err[(2)]
4	0.3	0.2	0.1	0.15			2	3	2			0.077	0.079	2.6	0.034	56.0
	0.1	0.3	0.4	0.2			2	4	4			0.086	0.086	0.0	0.064	25.6
	1.0	0.5	0.4	0.3			5	3	3			0.244	0.254	4.1	0.223	4.5
	0.5	0.3	0.3	0.2			5	4	5			0.181	0.185	2.2	0.174	3.9
	0.2	0.2	0.2	0.1			3	2	2			0.079	0.086	8.9	0.065	17.7
	0.3	0.3	0.3	0.1			4	2	3			0.091	0.092	1.1	0.089	2.2
	0.4	0.3	0.5	0.15			4	3	3			0.138	0.140	1.4	0.135	2.2
	0.5	0.4	0.6	0.15			3	3	3			0.143	0.145	1.4	0.140	2.1
	0.5	0.7	1.0	0.3			4	4	5			0.281	0.283	0.7	0.277	1.4
	0.3	0.4	0.5	0.1			3	4	3			0.097	0.098	1.0	0.097	0.0
5	0.3	0.4	0.3	0.3	0.1		2	1	1	1		0.064	0.075	17.2	0.022	66.0
	0.3	0.2	0.3	0.5	0.1		2	2	1	1		0.068	0.075	10.3	0.036	47.1
	0.5	0.3	0.4	0.5	0.3		2	2	2	2		0.177	0.200	13.0	0.167	5.6
	1.0	0.5	0.5	1.0	0.3		2	3	3	2		0.247	0.270	9.3	0.202	18.2
	1.0	1.5	1.0	1.5	0.5		2	2	3	1		0.352	0.375	6.5	0.232	34.0
	0.3	0.5	0.8	1.0	0.2		2	3	2	2		0.155	0.158	1.9	0.134	13.5
	1.5	1.0	1.5	1.0	1.0		2	3	3	2		0.594	0.667	12.3	0.083	86.0
	1.5	2.0	1.0	2.0	1.0		3	3	3	2		0.701	0.750	7.0	0.417	41.0
	2.0	1.5	1.0	1.5	1.0		3	3	3	3		0.709	0.750	5.8	0.437	38.4
	1.5	2.0	1.0	1.5	0.5		3	4	3	3		0.457	0.467	2.2	0.440	3.7
6	0.5	0.5	0.5	0.5	0.5	0.2	2	2	2	2	2	0.155	0.179	15.5	0.094	39.4
	0.3	0.3	0.3	0.4	0.4	0.4	2	2	2	2	2	0.156	0.227	45.5	-[(3)]	-
	1.0	1.5	1.0	0.5	0.15	0.3	2	2	2	2	2	0.236	0.245	4.2	0.187	20.8
	1.5	3.0	2.0	1.0	1.0	0.5	1	2	3	2	3	0.355	0.375	5.6	0.253	28.7
	1.5	1.0	2.0	1.5	1.3	0.7	2	3	2	2	2	0.511	0.589	15.3	0.255	50.1
	0.5	1.2	1.0	1.0	1.0	0.3	3	2	2	2	3	0.253	0.270	6.7	0.211	16.6
	1.5	1.5	2.0	3.0	1.5	1.0	2	2	3	2	3	0.691	0.789	14.2	0.312	54.8
	0.5	0.4	0.3	0.4	0.8	0.1	3	3	3	2	3	0.093	0.095	2.2	0.090	3.2
	0.4	0.3	0.6	0.2	0.5	0.3	2	2	3	2	3	0.144	0.158	9.7	-	-
	0.5	0.2	0.4	0.4	0.2	0.3	3	2	3	2	3	0.139	0.158	13.7	-	-

(1): (TUB-Exact)/Exact x 100

(2): (Exact-TLB)/Exact x 100

(3): meaningless bound (negative value)

Table 2. Throughput bounds for K-server simple ADQNs (K=6,7)

K	μ_1	μ_2	μ_3	μ_4	μ_5	μ_6	μ_7	B_1	B_2	B_3	B_4	B_5	B_6	Exact	TUB	%err(1)	TLB	%err(2)
6	1.5	1.0	1.5	1.5	2.0	0.5		1	2	2	1	2		0.323	0.375	16.1	0.116	64.1
	1.5	2.0	1.5	1.5	2.0	0.7		2	3	2	1	2		0.452	0.610	35.0	0.218	51.8
	1.5	2.0	1.5	1.5	2.0	0.5		2	3	3	2	2		0.433	0.462	6.7	0.381	12.0
	1.5	2.0	1.5	1.5	2.0	1.0		2	3	3	2	3		0.693	0.789	13.9	0.323	53.4
	1.0	1.5	2.5	2.5	3.0	1.0		2	4	3	3	3		0.660	0.667	1.1	0.487	26.2
	1.0	1.5	2.5	2.5	3.0	0.5		2	3	3	3	4		0.426	0.429	0.7	0.409	4.0
	2.5	2.0	1.5	1.5	2.5	0.5		2	2	2	1	2		0.365	0.462	2.7	0.280	23.3
	1.5	2.0	1.5	1.5	2.0	0.3		2	3	2	1	2		0.246	0.290	17.9	0.224	8.9
	1.5	2.0	2.5	2.5	2.0	0.5		2	3	3	2	2		0.446	0.462	3.6	0.413	7.4
	2.5	3.0	1.5	1.5	2.0	1.0		2	3	3	2	3		0.736	0.877	19.2	0.472	35.9
	2.0	2.0	3.0	1.5	2.0	1.0		2	1	2	3	3		0.633	0.667	5.4	0.257	59.4
	1.0	1.5	2.5	2.5	3.0	0.8		2	4	3	3	3		0.586	0.590	0.7	0.511	12.8
	1.5	2.0	2.5	3.0	2.5	1.0		2	2	2	2	2		0.716	0.789	10.2	0.365	49.0
	1.0	1.0	1.0	1.5	2.0	0.5		3	2	3	2	2		0.400	0.429	7.3	0.300	25.0
	3.0	2.0	1.0	2.0	2.0	0.7		4	3	2	2	3		0.530	0.543	2.5	0.444	16.2
	1.0	1.0	1.5	2.0	2.0	0.5		3	2	2	2	2		0.403	0.429	6.5	0.309	23.3
	2.0	3.0	1.5	1.5	2.0	1.0		3	3	3	2	4		0.749	0.877	17.1	0.542	27.6
	3.0	2.0	2.0	1.5	2.0	1.0		2	3	3	3	2		0.776	0.923	18.9	0.524	32.5
	2.0	2.5	3.0	2.5	3.0	1.0		2	3	2	4	4		0.825	0.857	3.9	0.717	13.1
	1.5	1.5	3.5	3.0	3.0	1.0		3	2	4	3	4		0.757	0.789	4.2	0.628	18.0
7	1.0	1.0	1.5	2.0	2.0	2.0	0.5	3	2	2	1	2	2	0.365	0.428	17.3	0.209	42.7
	1.5	2.0	1.5	1.5	2.0	1.0	0.3	2	3	2	2	1	1	0.221	0.290	31.2	0.162	26.7
	1.0	1.5	2.5	2.0	2.5	3.0	0.5	2	3	1	2	3	2	0.383	0.417	8.9	0.291	24.0
	1.0	2.0	2.5	1.5	2.0	2.5	0.3	2	2	2	3	1	2	0.256	0.281	9.8	0.228	11.0
	1.5	1.0	1.0	1.5	1.0	2.0	0.3	2	1	2	2	1	2	0.218	0.231	6.0	0.133	39.0
	1.0	1.5	2.5	2.0	2.5	3.0	0.7	2	3	2	2	3	2	0.521	0.543	4.2	0.358	31.3
	2.5	2.0	1.5	3.0	1.5	2.5	0.5	2	2	2	2	2	1	0.399	0.462	15.8	0.296	25.8
	1.0	1.0	1.5	2.0	2.0	2.0	0.5	3	2	2	2	2	2	0.400	0.429	7.3	0.285	28.8
	1.5	2.0	1.5	1.0	1.5	2.0	0.3	2	3	2	2	1	2	0.250	0.290	16.0	0.216	13.6
	1.5	2.0	1.5	1.5	1.5	2.0	0.5	2	3	2	2	2	2	0.415	0.462	11.3	0.316	23.9
	1.5	2.0	1.5	2.0	2.0	1.5	0.8	3	2	3	2	2	2	0.601	0.718	19.5	0.305	49.3
	1.5	2.0	1.0	2.0	2.0	3.0	0.5	2	3	2	3	1	2	0.386	0.429	11.1	0.283	26.7
	1.0	1.5	2.5	2.0	2.5	3.0	0.5	2	3	2	2	3	2	0.416	0.429	3.1	0.358	13.9
	1.0	2.0	2.5	1.5	2.0	2.5	0.3	2	2	2	3	2	2	0.274	0.281	2.6	0.257	6.2
	1.5	1.0	1.0	1.5	1.0	2.0	0.5	2	3	2	2	3	2	0.382	0.429	12.3	-.241	36.9
	2.5	1.5	2.5	1.5	2.5	3.0	1.0	2	3	2	2	2	3	0.752	0.877	16.6	0.369	50.9
	1.5	1.0	1.5	1.0	1.5	2.0	0.5	2	2	2	3	2	2	0.384	0.429	11.7	0.244	36.5
	1.5	1.5	1.5	1.0	2.0	2.5	0.5	3	2	2	2	3	2	0.403	0.462	14.6	0.312	22.6
	1.5	2.0	1.0	2.0	2.0	3.0	0.5	2	3	2	3	1	2	0.386	0.429	11.1	0.283	26.7
	1.0	2.0	2.5	1.5	2.0	2.5	0.5	2	2	2	3	2	3	0.416	0.429	3.1	0.354	14.9

(1): (TUB-Exact)/Exact x 100

(2): (Exact-TLB)/Exact x 100

Table 3. Throughput upper bounds for six-server acyclic ADQNs

K	μ_1	μ_2	μ_3	μ_4	μ_5	μ_6	B_1	B_2	B_3	B_4	B_5	Exact	TUB	%err
I	0.5	0.7	0.3	0.5	0.7	0.2	1	1	1	1	1	0.112	0.143	27.7
	0.3	1.0	0.5	1.0	0.2	0.6	1	2	1	2	2	0.150	0.185	23.3
	1.0	1.0	0.5	0.3	1.0	0.5	2	1	2	2	2	0.213	0.245	15.0
	1.0	1.0	0.8	1.0	0.2	1.5	2	2	1	1	3	0.160	0.167	4.3
	0.5	0.3	0.5	1.0	1.5	1.0	2	1	3	2	2	0.182	0.188	3.3
	1.5	0.5	1.0	1.0	1.0	2.0	2	2	3	2	1	0.410	0.429	4.6
	2.0	2.0	1.0	0.5	0.6	0.6	2	1	2	3	2	0.331	0.363	9.7
	0.3	1.0	0.5	1.0	0.5	1.0	2	3	3	2	2	0.242	0.245	1.2
	1.0	1.0	0.5	1.0	2.0	1.0	2	3	3	2	2	0.405	0.429	5.9
	1.0	0.5	0.5	0.8	2.0	1.0	2	2	3	3	3	0.321	0.333	3.7
	1.5	2.0	1.5	0.7	1.5	2.0	2	3	2	1	2	0.459	0.477	5.9
	1.0	2.0	3.0	1.0	1.5	1.0	2	1	2	3	3	0.647	0.750	15.9
	1.5	2.0	1.5	1.0	1.5	2.0	2	3	3	2	3	0.726	0.789	8.7
	1.0	1.5	2.5	1.0	2.5	3.0	2	4	3	3	3	0.766	0.833	8.7
	1.0	2.0	2.5	0.7	2.0	2.5	3	3	4	3	4	0.659	0.680	3.2
II	0.5	0.7	0.3	0.5	0.7	0.2	1	1	1	1	1	0.110	0.120	9.1
	0.3	1.0	0.5	1.0	0.2	0.6	1	2	1	2	2	0.157	0.175	11.5
	1.0	1.0	0.5	0.3	1.0	0.5	2	1	2	2	2	0.243	0.281	15.6
	1.0	1.0	0.8	1.0	0.2	1.5	2	2	1	1	3	0.162	0.167	3.1
	0.5	0.3	0.5	1.0	1.5	1.0	2	1	3	2	2	0.201	0.245	21.9
	1.5	0.5	1.0	1.0	1.0	2.0	2	2	3	2	1	0.398	0.429	7.8
	2.0	2.0	1.0	0.5	0.6	0.6	2	1	2	3	2	0.392	0.476	21.4
	0.3	1.0	0.5	1.0	0.5	1.0	2	3	3	2	2	0.251	0.270	7.6
	1.0	1.0	0.5	1.0	2.0	1.0	2	3	3	2	2	0.410	0.429	4.6
	1.0	0.5	0.5	0.8	2.0	1.0	2	2	3	3	3	0.345	0.429	24.3
	1.5	2.0	1.5	0.7	1.5	2.0	2	3	2	1	2	0.560	0.642	14.6
	1.0	2.0	3.0	1.0	1.5	1.0	2	1	2	3	3	0.621	0.750	20.8
	1.5	2.0	1.5	1.0	1.5	2.0	2	3	3	2	3	0.795	0.933	17.4
	1.0	1.5	2.5	1.0	2.5	3.0	2	4	3	3	3	0.707	0.789	11.6
	1.0	2.0	2.5	0.7	2.0	2.5	3	3	4	3	4	0.664	0.682	2.7

I: The network given in Fig. 2 (a)

II: The network given in Fig. 2 (b)

$$TH(\Sigma) \geq \mu_K[1-(\sum_{j\in U(K)} g(\mu_{u_j}, B_j, \mu_K) + \sum_{j\in D(K)} g(\mu_{d_j}, B_j, \mu_K))]$$

Proof.

$$TH(\Sigma) = \mu_K[1-P(\cup_{j\in U(K)} N_j=0 \ or \ \cup_{j\in D(K)} N_j=B_j)]$$

$$\geq [1-(\sum_{j\in U(K)} P(N_j=0) + \sum_{j\in D(K)} P(N_j=B_j))]$$

$$=[1-(\sum_{j\in U(K)} P(N_j=0) + \sum_{j\in U^r(K)} P(N_j^r=0))]$$

by Lemma 4

$$\geq [1-(\sum_{j\in U(K)} g(\mu_{u_j}, B_j, \mu_K) + \sum_{j\in U^r(K)} g(\mu_{u_j^r}, B_j, \mu_K))]$$

by Lemma 3

Since $U^r(K)=D(K)$ and $u_j^r=d_j$, $j\in U^r(K)$, the proof is completed. □

Remark. Note that our TLB→-∞ as μ_K is ∞, that is our lower bounding method will perform poorly for the case where $\mu_K \rangle \mu_j$, $j\in U(K)\cup D(K)$. But the TLBs become tighter for the case where $\mu_K \langle \mu_j$, $j\in U(K)\cup D(K)$.

4. Computational results

To show the effectiveness of the suggested throughput-upper and lower bounding methods, computational experiments are first conducted with ten input instances for each case of K-server simple assembly networks given in Fig 1 (a) (K=4,5,6) (Table 1) and twenty input instances for each case of K-server simple ADQNs given in Fig 1 (b) (K=6,7) (Table 2). Since there are no reported bounds or approximate solutions in the existing literature, comparison is made only with exact solutions.

As shown in Table 1 and Table 2, our TLB (Throughput Lower Bound) becomes looser as the ratio of arrival rate and service rate of the decomposed system becomes smaller, whereas our TUBs are very tight in most input instances. Table 3 summarizes the results of computational experiments conducted with fifteen instances for each of two six-server ADQNs given in Fig. 2 (a) and Fig. 2 (b). In these cases, we suggest only our TUBs because our lower bounding method does not perform any further for the networks other than the simple ADQNs.

The exact solutions shown in the tables are obtained by solving steady-state balance equations on HP 725 W/S with 32Mbytes memory. The complexity of this numerical method mainly depends upon the number of states ($\Pi_{i=1}^{M}(1+B_i)$), which limits the size of our experimental problems to those with the number of states below 3000. Moreover, experimental test indicates that the computational times for exact solutions increase explosively as the size of problem increases. For example, problems with more than 2000 states require several tens of hours. Considering the memory size and computational time for obtaining exact solutions, the size of experimental data is chosen. Note that our upper and lower bounding functions have the closed forms and hence the computational times for bounds are trivial.

5. Concluding Remarks

In this study, we have addressed throughput-upper and lower bounding methods for acyclic ADQNs with exponential service times. The throughput-bounding methods are based upon the monotonicity properties of network throughputs with respect to service times and buffer capacities. The extensive computational experiments indicate that the proposed bounding methods perform satisfactorily in the tightness of bounds. It may be worthwhile to note that our throughput-bounding methods are devised for only the throughput measure. Since the whole development of our paper has been dedicated to throughput-bounding methods, it would be of future research interest to develop an efficient approximate method.

References

[1] Altiok, T. and Kao, Z., "Bounds for throughput in

production/inventory system in series with deterministic processing times" *IIE Transactions* 21, 82-85, 1989.

[2] Ammar, M.H. and Gershwin, S.B., "Equivalence relations in queueing models of fork/join networks with blocking", *Performance Evaluation* 10, 233-245, 1989.

[3] Baccelli, F., Massey, W.A. and Towsley, D., "Acyclic fork-join queueing networks", *Journal of the ACM* 36 (3), 615-642, 1989.

[4] Bhat. U.N., "Finite capacity assembly like-queues", *Queueing Systems* 1, 85-101, 1986.

[5] Bonomi F., "An approximate analysis for a class of assembly-like queues", *Queueing Systems* 1, 289-309, 1987.

[6] Dallery Y. and Frein, Y., "On decomposition method for tandem queueing networks with blocking", *Operations Research* 41 (2), 386-399, 1993.

[7] Gershwin, S.B., "An efficient decomposition method for the approximation evaluation of tandem queues with finite storage space and blocking", *Operations Research* 35 (2), 291-305, 1987.

[8] Gershwin, S.B., "Assembly/disassembly systems: an efficient decomposition algorithm for tree-structured networks", *IIE Transactions* 23 (4), 302-314, 1991.

[9] Hopp, W.J. and Simon, J.T., "Bound and heuristics for assembly-like queues", *Queueing Systems* 4, 137-156, 1989.

[10] Lipper, E.H. and Sengupta, B., "Assembly-like queues with finite capacity: bounds, asymptotics and approximations", *Queueing Systems* 1, 67-83, 1986.

[11] Liu, Y.C. and Perros, H.G., "A decomposition procedure for the analysis of a closed fork/join queueing systems", *IEEE Transactions on Computers* 40 (3), 365-370, 1991.

[12] Onvural, R.O. and Perros, H.G., "On equivalencies of blocking mechanisms in queueing networks with blocking", *Operations Research Letters* 5(6), 293-297, 1986.

[13] Paik, C.H. and Tcha, D.W., "Throughput equivalencies in fork/join queueing networks with finite buffers and general service times", *International Journal of Production Research* 33(3), 695-703, 1995.

[14] Ross, S.M., *Stochastic Processes*, John Wiley & Sons, New York, 1983.

[15] Shanthikumar, J. G. and Jafari, M. A., "Bounding the performance of tandem queues with finite buffer spaces", Technical Report, School of Business Administration, University of California, Berkeley, CA, 1987.

[16] Stoyan, D., *Comparison methods for queues and other stochastic models*, John Wiley and Sons, 1983.

[17] Tcha, D.-W., Paik, C.H. and Lee, W.T., "Throughput upper bounds for open Markovian queueing networks with blocking", *Computer ind. Engng.* 28(2), 351-365, 1995.

97년 7월 최초 접수, 97년 10월 최종 수정