View metadata, citation and similar papers at core.ac.uk

brought to you by CORE

provided by Purdue E-Pubs

Purdue University

Purdue e-Pubs

Department of Computer Science Technical Reports

Department of Computer Science

1988

A Band and Bound Technique for Simple Random Algorithms

Vernon J. Rego
Purdue University, rego@cs.purdue.edu

Report Number:
88-844

Rego, Vernon J., "A Band and Bound Technique for Simple Random Algorithms" (1988). *Department of Computer Science Technical Reports.* Paper 721.
https://docs.lib.purdue.edu/cstech/721

This document has been made available through Purdue e-Pubs, a service of the Purdue University Libraries.
Please contact epubs@purdue.edu for additional information.

A BAND AND BOUND TECHNIQUE FOR SIMPLE RANDOM ALGORITHMS *

Vernon J. Rego

Department of Computer Sciences
Purdue University
West Lafayette, IN 47907

* This research was supported in part by NSF under grant NCR-8702115.

1. INTRODUCTION

A simple random algorithm $\mathcal{A}$ is an algorithm whose behavior is associated with a first order Markov chain [1,2]. The algorithm $\mathcal{A}$ traverses a sequence of "states" during the course of its execution, where a state is defined by variables the algorithm uses to record its temporal progress. For example, $\mathcal{A}$ may be a probabilistic finite state automaton, in which case $\mathcal{A}$ makes a transition by reading an input and consulting a probability transition table before moving randomly from a current state to a target state. In this case each state is given by a tuple made up of an input and a state of the finite state automaton. We generally assume that the input is such that $\mathcal{A}$ terminates in a finite time. Correspondingly, the Markov chain representing $\mathcal{A}$ is a transient chain with one or more absorbing states.

Let $\mathcal{A}$ be a random algorithm whose states can be mapped onto the nonnegative integers $Z = \{0,1,2,\ldots\}$, with either a finite or countable number of states. Given specific information concerning the behavior of $\mathcal{A}$ and its input, we are interested in $\mathcal{A}$'s expected run time, i.e., the number of steps required for the algorithm to terminate. The standard approach in computing $\mathcal{A}$'s expected run time is *numerical* [1,2] since $\mathcal{A}$'s behavior is governed by a Markov chain. Briefly, let $P(\mathcal{A})$ be the transition probability matrix of the Markov chain, and assume there is a single absorbing state, say state 0. If there is more than absorbing state, then these states can all be lumped together into a single absorbing state. This is done by adding the probabilities corresponding to the absorbing states in each row of $P(\mathcal{A})$ and placing the sum in the column position corresponding to the single (lumped) absorbing state. Order the rows (and columns) of $P(\mathcal{A})$ in decreasing order of state index, let M be the submatrix obtained by deleting the row and column in $P(\mathcal{A})$ corresponding to the absorbing state. If we begin to execute algorithm $\mathcal{A}$ in state n, then its expected run time is given by

$$E[T_n(\mathcal{A})] = \delta_n (I - M)^{-1} e \qquad (1.1)$$

where δ_n is a vector with a 1 in the n^{th} coordinate (counting from right to left) and zeros else-

where, e is a vector with all entries equal to 1, and $(I - M)^{-1}$ is the fundamental matrix of the chain.

It is of some interest to determine, qualitatively speaking, if one can get a handle on $E[T_n(\mathcal{A})]$ without resorting to Eq. (1.1). In other words, is it possible to estimate the expected run time of a simple random algorithm *non-numerically*, by either examining the structure or utilizing some property of $\mathbf{P}(\mathcal{A})$? Interestingly enough, the answer is in the affirmative. In the following sections we examine how some simple upper and lower bounds on $E[T_n(\mathcal{A})]$ may be obtained by imposing certain small requirements on $\mathbf{P}(\mathcal{A})$. The bounds are of type $O(\log n)$ and may help give insights into why certain algorithms yield logarithmic expected run times. Additionally, this type of analysis can also be useful in the expected run time estimation of some deterministic algorithms (see section 4). Examples of algorithms which can be viewed as simple random algorithms are given in section 2. It is shown that expected run times for deterministic algorithms can be estimated by placing distributional assumptions on the algorithm's input and focusing on a Markov chain that is constructed to represent the expected behaviour of the algorithm. In section 3, we present the main results, specifying conditions under which an algorithm $\mathcal{A}$ yields

$$c_1 \log_\alpha n \;\leq\; E[T_n(\mathcal{A})] \;\leq\; c_2 \log_\beta n \qquad (1.2)$$

for inputs depending on n and certain constants c_1, c_2, β, α with $\beta \geq \alpha > 1$. In each case, α and β are parameters of model and can usually be determined by examining the transition probability matrix of an appropriate Markov chain. Finally, in section 4 we apply the ideas of section 3 to the examples presented in section 2.

2. SOME SIMPLE RANDOM ALGORITHMS

2.A Feedback-less broadcast protocols [Hofri, 1987]

Consider a communications network with n distributed transmitting (and receiving) stations. Divide the time axis into equal sized slots and assume that transmitters are synchronized with respect to slot boundaries. A *phase* of the protocol comprises a random number of slots and is defined as follows. Initially (at the beginning of slot 0) all n stations are *active* (i.e., capable of transmitting with probability 1). At the end of each slot j, $j \geq 0$, each active station transmits a message and either becomes inactive hereafter, with probability p, or remains active with probability $q = 1 - p$. To avoid trivialities, we take $0 < p < 1$. The intention is to allow one particular node to receive necessary information from neighbors who possess such information. A *phase* terminates if, at any slot, only a single station transmits (i.e., a successful phase) or all stations become inactive (i.e., an unsuccessful phase). What is the expected length of a phase for large n ? Suppose that we allow each station i to transmit with probability p_i if it is active, with $p_i \neq p_j$ for $i \neq j$. In this case, can we still find bounds on the expected length of a phase ?

2.B Maximal Independent Sets in Random Graphs

Let G be a random graph on n nodes generated as follows. For each two tuple (i,j), $i < j \leq n$, $1 \leq i \leq n - 1$, let $E_{i,j}$ be a Bernoulli random variable with parameter p for which an instance is observed. If event $\{E_{i,j} = 1\}$ occurs, then place an edge between nodes i and j and call nodes i and j *adjacent* nodes.

Given a random graph G, the lexicographically first maximal independent set (LFMIS) is generated by a very simple algorithm. First initialize the set S to be the empty set. Next, for $i = 1,2, \ldots, n$, if node i is *not* adjacent to any node in S, then add node i to the set S. What is the expected number of steps required to generate S ? As in the previous example, $E_{i,j}$ can be

generalized to a Bernoulli random variable with parameter $p_{i,j}$, for $1 \le j \le n$. How does this affect the expected run time of the LFMIS algorithm ?

2.C Sequential Searches [Knuth, 1973]

We are given a table of records $R_1, R_2, \ldots, R_n$, with respective keys $K_1, K_2, \ldots, K_n$. We are interested in locating a record with a specified key, say K. For generality, assume that K resides in position i with some unknown probability p_i so that $\sum_{i=1}^{n} p_i = 1$ (i.e., search will always be successful). Clearly, an unsuccessful search requires a scan of the entire list. What is the expected number of comparisons required for a successful search ?

2.D Random sorts

This example demonstrates an unusual nondeterministic sorting scheme on a uniprocessor. The algorithm was devised in order to help analyze a similar distributed sorting scheme on n processors. We begin with the uniprocessor sort. Given an input list L of n integers, the task is to sort the list in ascending order. Without loss of generality, assume that $1 \le L[j] \le M$, for $1 \le j \le n$ and some arbitrary integer M.

Consider an algorithm which begins with list $L = L^{(0)}$ and progresses in nondeterministic fashion through a sequence of lists $L^{(1)}, L^{(2)}, \ldots, L^{(k)}, \ldots$ until finally terminating with the required sorted list $L_S = L^{(m)}$. At each step, k, $0 \le k \le m$, the algorithm examines the current list $L^{(k)}$ and computes the number of *flags* present. Position j in any list L is said to generate a *flag* if $L[j] > L[j+1]$ for $1 \le j \le n-1$. Let $F(k)$ denote the number of flags present in list $L^{(k)}$. In order to obtain list $L^{(k+1)}$, the algorithm randomly chooses one of the flagged positions in $L^{(k)}$ and exchanges the flagged element with its right neighbor. Each of the $F(k)$ positions is equally likely to be chosen. The algorithm terminates with some list $L^{(m)}$ provided $F(m) = 0$ (i.e., all flags have been eliminated). Clearly, the simple random algorithm just described may

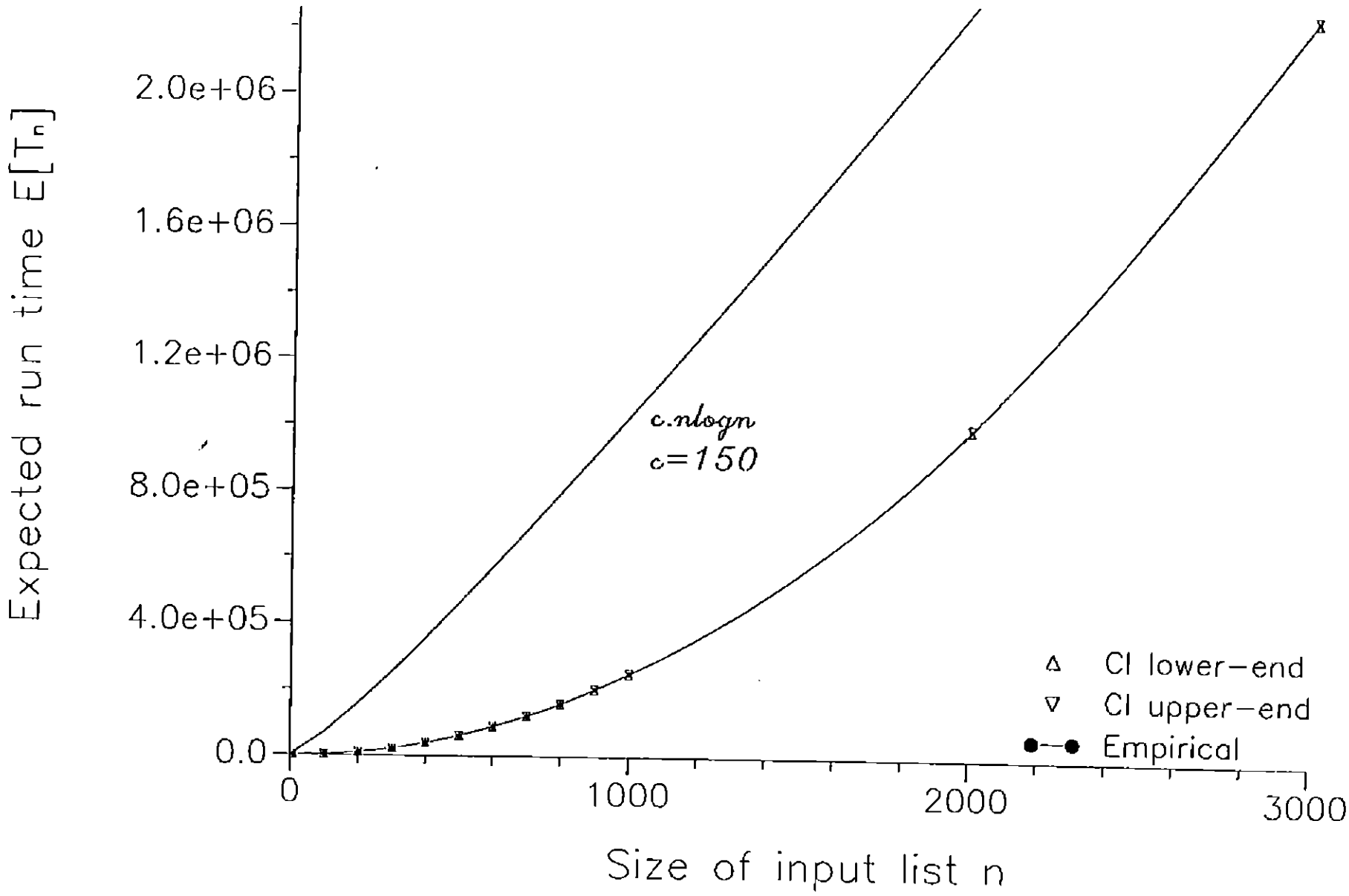

Fig.1 Expected Time for Distributed Sort

have an extremely large sorting time (i.e., number of comparisons). The question is can we find a bound on its expected run time ?

Consider how we could generalize the idea to a distributed sorting scheme if n processors were available. Initially, load each processor j with list element $L[j]$, for $1 \le j \le n$. We say that processor $(j+1)$ is the right neighbor of processor j, for $1 \le j \le n-1$ (and note that processor n has no right neighbor). Additionally, assume that:

(1) processors communicate asynchronously,

(2) only processors with flagged elements are allowed to initiate exchanges,

(3) simultaneous initiations of exchanges are illegal, and

(4) a processor with a flagged element can initiate an exchange only with its right neighbour.

The distributed sorting scheme works as follows. Any processor that finds its right neighbour possessing an element smaller than its own will initiate an exchange to restore local order. Since only one initiation can occur at a time, an error condition in which a processor receives an incorrect element will not arise. After a finite number of such exchanges each processor will have obtained its final list element and will stop initiating exchanges so that the algorithm terminates. A simulation model of the sorting scheme was developed and some empirical estimates of its expected run time are shown in Figure 1. Can we analytically estimate the expected run time of the distributed sort ?

2.E Dependent service queueing systems [Neuts, 1977]

A queueing system can be viewed as an extension of a simple random algorithm in the sense that, given an initially nonempty queue, the server has to execute a simple algorithm (keep serving customers) until the queue eventually becomes empty. A stable queue ensures that the algorithm terminates in a finite time. While such a view of a queue may be unusual, it is natural to use the tool we develop towards such an application simply due to the existence of an

underlying Markov chain, albeit one with remarkable structure.

Consider a single server queueing system in which customer arrivals occur in groups at the epochs of a homogeneous Poisson process of rate λ. The sizes of successive groups are stochastically independent and identically distributed with density $\{p_k, k \geq 1\}$. Assume that customer types and service times are generated by an m-state Markov renewal sequence with an irreducible transition probability matrix. Such a queue with customers whose service times are *dependent* is useful in modeling buffers in computer networks [5]. The so-called batch $M \mid SM \mid 1$ queue [4] yields a customer type and queue length embedded Markov chain

$$Q = \begin{bmatrix} B_0 & B_1 & B_2 & B_3 & \cdot & \cdot & \cdot & \cdot \\ A_0 & A_1 & A_2 & A_3 & \cdot & \cdot & \cdot & \cdot \\ 0 & A_0 & A_1 & A_2 & \cdot & \cdot & \cdot & \cdot \\ 0 & 0 & A_0 & A_1 & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & & \cdot & & \\ \cdot & \cdot & \cdot & \cdot & & \cdot & & \\ \cdot & \cdot & \cdot & \cdot & & \cdot & & \\ \cdot & \cdot & \cdot & \cdot & & \cdot & & \end{bmatrix} \tag{2.1}$$

where B_j and A_j are substochastic matrices generated by the model, for $j \geq 0$. The matrix Q describes a stochastic process $\{(X_k, Y_k), k \geq 0\}$ at customer departure epochs, where X_k is the queue size and Y_k is the customer type at departure instants. Given that the queue initially contains n customers, can we obtain a bound on the expected number of customers served before the queue becomes empty ? The known methods for computing such a quantity (i.e., a passage time) are strictly numerical [4]. In section 4, we demonstrate how a simple bound on this quantity can be obtained without resorting to computation.

3. ON BANDING AND BOUNDING

Let $\mathcal{A}$ be a simple random algorithm with underlying Markov chain $\{Z_k ; k \geq 0\}$ defined on a countable set of states $\mathbf{Z} = \{0,1,2, \ldots\}$. For ease of notation, we will focus our attention on $\{Z_k\}$ and neglect mention of $\mathcal{A}$. Assume that state 0 is the only absorbing state for the chain $\{Z_k\}$ (for otherwise we can lump the absorbing states together into a single state, as explained in section 1). Given that $Z_0 = n$, i.e., we start the chain in state n, let T_n denote the number of steps (run time) required by the chain before it reaches state 0. Though we may work with a countable state space, we assume that we only deal with chains for which $\lim_{m \to \infty} P(Z_m = 0) = 1$. We begin by demonstrating a sufficient condition for $E(T_n)$ to have an $O(\log n)$ upper bound. The argument used to obtain the result appeals to a negative drift requirement for absorbing Markov chains.

Theorem 1

Let $\{Z_k ; k \geq 0\}$ be a Markov chain defined on the set Z, with 0 as an absorbing state and all other states transient. If the inequality

$$E(Z_{j+1} \mid Z_j = m) \leq \frac{m}{\beta} \tag{3.1}$$

is satisfied for all m, $m \geq 1$, and an arbitrary $\beta > 1$, then

$$E(T_n) \leq \log_\beta n + \frac{\beta}{\beta - 1} \tag{3.2}$$

Proof: Using P_n and E_n to denote probability and expectation, respectively, conditional on $Z_0 = n$, inequality (3.1) yields

$$E_n(Z_{j+1} \mid Z_j) \leq Z_j \cdot \frac{1}{\beta}$$

and the recursion

$$E_n(Z_{j+1}) \leq E_n(Z_j) \cdot \frac{1}{\beta}$$

from which is obtained

$$E_n(Z_j) \le n \cdot \frac{1}{\beta^j} \tag{3.3}$$

Since $P_n(Z_j \neq 0) \le E_n(Z_j)$, we get

$$P(T_n > j) \le E_n(Z_j) \le \min(\frac{n}{\beta^j}, 1) \tag{3.4}$$

Finally, using the relation $E(T_n) = \sum_{j \ge 0} P(T_n > j)$, we obtain

$$E(T_n) \le \sum_{j=0}^{\log_\beta n} 1 + \frac{1}{\beta - 1} \tag{3.5}$$

and the result

$$E(T_n) \le \log_\beta n + \frac{\beta}{\beta - 1}$$

□

A version of the above theorem was used by Stavskaya and Pyateskii-Shapiro [6] to prove existence of an invariant measure in a Markov random field for parameter values above a certain threshold. The version was attributed to L.G. Mituscin, but no proof of the theorem was given. Additionally, the linear term shown in [6] is $\frac{1 + \ln \beta}{\ln \beta}$, which is larger than our linear term $\frac{\beta}{\beta - 1}$.

A more general version of Theorem 1 can easily be shown using essentially the same argument. Define $f: \mathbf{Z} \to \mathbf{Z}$ to be some increasing function with $f(0) = 0$. By replacing condition (3.1) of Theorem 1 by the condition

$$E[f(Z_{j+1}) \mid Z_j = m] \le \frac{f(m)}{\beta} \tag{3.6}$$

for all m, $m \ge 1$ and an arbitrary $\beta > 1$, we can repeat the arguments used in Theorem 1 to obtain

Corollary 1.1.

Let $\{Z_k\}$ be a Markov chain on $\mathbf{Z}$ with 0 as an absorbing state and all other states transient. If

$\{Z_k\}$ satisfies (3.6), then

$$E[T_{f(n)}] \leq \log_\beta f(n) + \frac{\beta}{\beta - 1} \tag{3.7}$$

□

Consider the following examples for function $f(\cdot)$.

(i) With $f(j) = j$, $j \in \mathbf{Z}$, one obtains $E(T_n) \sim O(\log n)$, since the corollary reduces to Theorem 1.

(ii) With $f(j) = j^2$, $j \in \mathbf{Z}$, one obtains

$$E(T_{n^2}) \leq 2\log_\beta n + \frac{\beta}{\beta - 1} \sim O(\log n).$$

(iii) With $f(j) = j^j$, $j \in \mathbf{Z}$, one obtains

$$E(T_{n^n}) \sim O(n \log n).$$

(iv) The function $f(j) = \theta^j$, for $j \in Z$, and $\theta > 1$ yields

$$E(T_{\theta^n}) \leq n \log_\beta \theta + \frac{\beta}{\beta - 1} \sim O(n).$$

(v) A special choice of $f(\cdot)$ yields a stronger form of Mituscin's result [6]. Let $\mathbf{Z}$ be decomposed and ordered into mutually disjoint sets $S_0 = \{0\}, S_1, S_2, \ldots,$ etc., so that $f(j) = k$ if $j \in S_k$, for $k, j \in \mathbf{Z}$. Then, using Corollary 1.1, the expected time $E[T_n]$ to reach state 0, starting from any state in set S_n, is given by (3.7).

□

We next take a look at a motivating and illustrative example of the use of Theorem 1. Consider a set of n urns labeled 1 through n arranged in a row. Inside each urn is placed a single ball. At each time step k, $k = 1,2,3, \ldots,$ we pick up a ball from each *nonempty* urn and toss it into the air. The tossing is done simultaneously for all the nonempty urns. Assume that for each nonempty urn, the tossed ball falls out of the urn with probability $p > 0$, and falls back into the urn with probability $q = 1 - p$. We call this algorithm an *urn game* with parameters

(n, q) and seek an estimate of the number of steps required before all n urns are found empty.

Let Z_k be the number of nonempty urns remaining after the k^{th} toss. With initial state $Z_0 = n$, the sequence $\{Z_k ; k \geq 1\}$ is clearly a Markov chain, with transition probabilities

$$P(Z_{k+1} = j \mid Z_k = i) \;=\; \binom{i}{j} p^{i-j} q^j \tag{3.8}$$

for $1 \leq i \leq n$, $0 \leq j \leq i$. Observe that the transition probability matrix for this chain is upper triangular and the probabilities in row i define a distribution which is binomial (n, q). That is, $\mathbf{P}_{i,j}$ is given by (3.8), for $1 \leq i \leq n$, $0 \leq j \leq i$, and state $i = 0$ absorbing, where $\mathbf{P}$ is the transition probability matrix of chain $\{Z_k\}$. Given that we start the urn game with n nonempty urns, then for each state i,

$$E(Z_{k+1} \mid Z_k = i) \;=\; i\,q \tag{3.9}$$

for $1 \leq i \leq n$. Since (3.9) demonstrates that the hypothesis of Theorem 1 is satisfied, we immediately obtain

$$E(T_n) \;\leq\; \frac{\log n}{\log q^{-1}} + \frac{1}{1-q} \tag{3.10}$$

or asymptotically, $E(T_n) \sim O(\log n)$.

A natural question to be asked at this stage is whether we can establish a lower bound in the same spirit as Theorem 1. While we cannot obtain a lower bound of the form in (3.2), and indeed cannot even use the arguments of Theorem 1, the urn model can be used to demonstrate that such a lower bound must hold for a large class of Markov chains, and thus for a large class of simple random algorithms.

To demonstrate the existence of a lower bound, notice that the probability that a ball falls out of its urn within k steps equals $1 - q^k$. Using $P(k,n)$ to denote the probability that all n urns are empty at step k, clearly

$$P(k,n) \;\leq\; (1-q^k)^n. \tag{3.11}$$

Since $P(k,n)$ is a nondecreasing function of k, it follows that

$$E(T_n) = \sum_{k=0}^{\infty} [1 - P(k,n)] \tag{3.12}$$

$$\geq k[1 - P(k,n)]$$

$$\geq k[1 - (1 - q^k)^n]$$

for any $k \geq 0$. Choosing

$$k = \frac{(1-\delta)\log n}{\log q^{-1}} \tag{3.13}$$

for $0 < \delta < 1$, we obtain $q^k = n^{\delta-1}$, and

$$E(T_n) \geq (1-\delta)(1-\varepsilon)\frac{\log n}{\log q^{-1}} \tag{3.14}$$

for any $\varepsilon > 0$, $0 < \delta < 1$, and $n > n_0$ so that

$$(1 - n_0^{\delta-1})^{n_0} < \varepsilon. \tag{3.15}$$

This suggests the following corollary..

Corollary 1.2.

Let $\{Z_k; k \geq 0\}$ be a Markov chain on $\{0, 1, 2, \cdots, n\}$, with transition probabilities given by (3.8) for $1 \leq i \leq n$, $0 \leq j \leq i$, and state 0 an absorbing state. With $Z_0 = n$, let T_n denote the time to absorption in this chain. Then

$$\lim_{n\to\infty} \inf \frac{T(n)}{\log_q n} \geq C \tag{3.16}$$

for some constant C.

□

Given a simple random algorithm with underlying Markov chain $\{Z_k\}$, our chief goal is to obtain an estimate of its expected run time. If we can *compare* the chain $\{Z_k\}$ in the sense of time to absorption to a (different) chain that satisfies the conditions (3.1), (3.6) or the hypothesis of Corollary 1.2, then we will be able to establish bounds on its expected run time. A useful tool for such a comparison is the notion of a stochastic order relation. A random variable A is

said to be *stochastically larger* than a random variable B if

$$P\{A > x\} \geq_{st} P\{B > x\} \tag{3.17}$$

for all x. Inequality (3.17) is usually written as $A \geq_{st} B$. Using this tool we now examine how the running times of two different algorithms can be compared, given that we know their underlying chains.

Let $\{A_k\}$ and $\{B_k\}$ be two Markov chains defined on $\mathbf{Z}$ such that both chains have state 0 as an absorbing state and all other states are transient. Let T_n^A and T_n^B denote the times to absorption from state n for chains $\{A_k\}$ and $\{B_k\}$, respectively. If one process exhibits sample paths that tend to lie *above* the sample paths of the other process, then the latter process will possess a smaller mean time to absorption.

Theorem 2:

If $A_0 = B_0 = m$ and $A_1 \geq_{st} B_1$ for all m, $m \geq 1$, then

$$E(T_n^A) \geq E(T_n^B) \tag{3.189}$$

for any initial state n.

Proof: By our hypothesis, we can begin with $A_0 = B_0 = n$. Since $A_1 \geq_{st} B_1$, we must have, for any increasing function $h(\cdot)$,

$$h(A_1) \geq_{st} h(B_1).$$

We choose, in particular, the function

$$h(A_1) = P_{A_1}\{A_{j-1} \geq k\}, \tag{3.19}$$

so that

$$E_n[P_{A_1}\{A_{j-1} \geq k\}] \geq E_n[P_{B_1}\{B_{j-1} \geq k\}]$$

implying that

$$E_n[P_n\{A_j \geq k\}] \geq E_n[P_n\{B_j \geq k\}]$$

and

$$P_n\{A_j \geq k\} \;\geq\; P_n\{B_j \geq k\}. \tag{3.20}$$

Using $k = 1$ in inequality (3.20), and observing that $\{A_j \geq 1\} \equiv \{T_n^A > j\}$, we obtain

$$P(T_n^A > j) \;\geq\; P(T_n^B > j) \tag{3.21}$$

for all j, $j \geq 0$. Since $T_n^A \geq_{st} T_n^B$, it follows that $E(T_n^A) \geq E(T_n^B)$.

□

At this stage, it is helpful to take stock of what we have done so far. Given a Markov chain $\{Z_k\}$ on $\{0,1,2,\ldots\}$ so that 0 is an absorbing state and all other states are transient, we have determined bounds on the expected time to absorption of the chain starting from any initial state n. That is, if T_n^Z denotes the time to absorption from state $Z_0 = n$, then

(a) if (3.1) is satisfied, $E(T_n^Z) \sim O(\log n)$,

(b) if (3.6) is satisfied, $E(T_{f(n)}^Z) \sim O(\log f(n))$.

The above results rely on expectation conditions. To get around this for a lower bound, we introduced the n-urn game Markov chain $\{U_k\}$ with parameter $q = 1 - p > 0$ and determined that

(c) $E(T_n^U) \sim O(\log n)$, and

(d) $E(T_n^U) \sim \Omega(\log n)$.

Finally, Theorem 2 tells us what conditions are required of Markov chains $\{X_k\}$, $\{Y_k\}$ so that

(e) $E(T_n^Y) \;\geq\; E(T_n^X)$.

A glance at (a) through (d) demonstrates a *bounding* technique for simple random algorithms. If one can show that a random algorithm possesses an underlying chain with any of the specified properties, or if the chain can be compared to another chain whose absorption time is known, then a bound for the expected run time of the random algorithm in question immediately follows. Condition (e) allows us to go a step further. If we can place the underlying

Markov chain, in the sense of time to absorption, *in between* two other Markov chains whose respective times to absorption are known, then we simultaneously obtain both upper and lower bounds for the expected run time of the random algorithm in question. This is essentially a *banding* technique. In order to demonstrate the idea, we draw upon a result from [7] without reproducing the proof.

Corollary 2.1.

Let $\{U_k ; k \geq 0\}$ be an urn game on n urns with parameter $q = 1 - p > 0$. Then

$$E(T_n^U) = \frac{\log n}{\log q^{-1}} + \frac{\gamma}{\log q^{-1}} + \frac{1}{2} + P(n) + O(n^{-1}) \tag{3.22}$$

where γ is Euler's constant, and $P(n)$ is a periodic function with very small amplitude.

In [7] the above result is obtained using purely analytic (asymptotic) methods. The arguments in Theorem 1 and Corollary 1.2 yield essentially the same asymptotics, but with simple, constructive arguments. Additionally, as we shall see in section 4, the construction enables us to obtain bounds for a large class of random algorithms.

Suppose that we are given an absorbing Markov chain $\{Z_k\}$ and one is able to show that the sample paths of $\{Z_k\}$ tend to lie above and below the sample paths of two other chains $\{X_k\}$ and $\{Y_k\}$, respectively. Theorem 2 tells us that the mean absorption time $E(T_n^Z)$ should satisfy

$$E(T_n^X) \leq E(T_n^Z) \leq E(T_n^Y). \tag{3.23}$$

The above result is what we call a banding argument and is stated more precisely for chains $\{X_k\}$ and $\{Y_k\}$ whose behavior we understand.

Theorem 3:

Let $\{X_k ; k \geq 0\}$ and $\{Y_k ; k \geq 0\}$ be urn games on n urns with parameters $q_X = 1 - p_X > 0$ and $q_Y = 1 - p_Y > 0$, respectively. Let $\{Z_k ; k \geq 0\}$ be a Markov chain on $\mathbb{Z}$ with state 0 as an absorbing state and all other states transient. If

$$E(X_1 \mid X_0 = m) \;\le\; E(Z_1 \mid Z_0 = m) \;\le\; E(Y_1 \mid Y_0 = m) \tag{3.23}$$

holds for all $m \ge 1$, then

$$c_X(\log n + \gamma) + g_X(n) \;\le\; E(T_n^Z) \;\le\; c_Y(\log n + \gamma) + g_Y(n) \tag{3.24}$$

where, for $x \in \{X,Y\}$, $c_x = (1 - p_x)^{-1}$, and

$$g_x(n) = \frac{1}{2} + P(n) + O(n^{-1}).$$

Proof: Since (3.23) guarantees the hypothesis of Theorem 2, it readily follows that

$$E(T_n^X) \;\le\; E(T_n^Z) \;\le\; E(T_n^Y)$$

$\forall\, n \ge 1$. Inequality (3.24) results from an application of Corollary 2.1.

□

It must be noted that $g_X(n)$ and $g_Y(n)$ approach the constant $\frac{1}{2}$ for large n. This is because $\lim_{n\to\infty} \frac{1}{n} = 0$, and $\lim_{n\to\infty} P(n)$ is negligible (Knuth [3] shows that $P(n) < 1.725 \times 10^{-7}$ for $q = \frac{1}{2}$).

4. APPLICATIONS

In order to demonstrate how the simple techniques developed in Section 3 can be used, we apply them to the examples given in Section 2. In each case we begin by formally describing the algorithm's underlying Markov chain and then use the appropriate result to obtain bounds on its expected run time.

4.A Feedback-less broadcast protocols

Let $\{U_k; k \ge 0\}$ be a sequence of random variables, where U_k is the number of active stations at the beginning of slot k. With $U_0 = n$, and a constant value of $q = 1 - p$, $\{U_k\}$ is a time homogeneous Markov chain on the set $\{0,1,2,\ldots,n\}$. Let T_n^U denote the length of a phase of the protocol, with $T_n^U = m$ if $U_{m-1} \notin \{0,1\}$ and $U_m \in \{0,1\}$. Clearly, $\{U_k\}$ is an urn game with parameters (n, q).

From Theorem 1, it follows that

$$E(T_n^U) \leq \log_\beta n + \frac{\beta}{\beta - 1} - O(\log n) \tag{4.1}$$

for $\beta = q^{-1}$, since we can properly *bound* the expected value of each row of the probability transition matrix of the chain. Additionally, since we can *stochastically* bound the state U_1, given $U_0 = m$ for any m, by identifying $\{U_k\}$ with an urn game, Corollary 1.2 gives

$$E(T_n^U) - \Omega(\log n). \tag{4.2}$$

In order to further refine our estimate of $E(T_n^U)$, we resort to Corollary 2.1 to obtain

$$E(T_n^U) - \log_\beta n + \frac{\gamma}{\log \beta} + \frac{1}{2} + P(n) + O(n^{-1}). \tag{4.3}$$

If each (active) node i, $1 \leq i \leq n$, has a probability p_i of remaining active at the end of a slot, then a further refinement of (4.3) is nontrivial. However, if we set $q = \max \{ q_i \mid i \leq i \leq n \}$, then our bounding arguments still yield the bounds (4.1) and (4.2). This is equivalent to saying that the expected length of a phase (urn game) in a system with asymmetric probabilities of transmission is determined by the station (urn) whose transmission (ball) has the largest probability of remaining active (falling back into the urn) at each step.

4.B Maximal Independent Sets in Random Graphs

Given a random graph (the number of steps required to obtain one is $O(n^2)$), we seek the average number of steps that must be executed by the LFMIS algorithm described in section 2. The algorithm described in 2.B proceeds by constructing a sequence of sets $S_0 = \phi$, $S_1, S_2, \ldots, S$, where S is the required LFMIS.

Consider the construction of some intermediate set S_i. Node i is compared to each of the elements in set S_{i-1}. If found adjacent to any of these elements (the probability of this is p in each case), node i is discarded, and otherwise S_i is given by $S_{i-1} \cup \{i\}$. This comparative behavior in the construction of S_i can be represented by a Markov chain $\{V_k(i)\}$ on the states $\{0, 1, 2, \ldots, i-1\}$, provided we make the (worst case) assumption that S_{i-1} contains the

elements $\{1,2,3,\ldots,i-1\}$. We demonstrate a logarithmic bound in spite of such an assumption. The underlying chain $\{V_k(i)\}$ behaves as follows. Starting with $V_0(i) = i$, the algorithm stops (i.e., $V_1(i) = 0$) if node i is found adjacent to node $(i-1)$, else the chain moves to state $(i-1)$ (i.e., $V_1(i) = (i-1)$) so that a check for adjacency to node $(i-2)$ can be made. The transition probability matrix of the chain is given by

$$\mathbf{P}_{V(i)}[j,k] = \begin{cases} q & k = j-1 \\ p & k = 0 \end{cases} \tag{4.4}$$

for $2 \le j \le i$, and $\mathbf{P}_{V(i)}[j,0] = 1$ for $j = 0, 1$. Theorem 1 yields

$$E[T_i^{V(i)}] \le \log_\beta(i-1) + \frac{\beta}{\beta-1}. \tag{4.5}$$

Since the procedure is repeated for each node i, $1 \le i \le n$, the average number of steps required for the LFMIS algorithm is bounded from above by $E(T_n^V)$, where

$$\begin{aligned} E(T_n^V) &= 1 + \sum_{i=2}^{n} E(T_{i-1}^{V(i)}) \\ &\le 1 + \sum_{i=2}^{n} \left[\log_\beta(i-1) + \frac{\beta}{\beta-1}\right] \\ &= \log_\beta(n-1)! + O(1) \\ &= c_1(n-1)\log_2(n-1) - c_2(n-1) + O(\log n) \end{aligned} \tag{4.6}$$

where $c_1 = (\log_2\beta)^{-1}$ and $c_2 = (\ln 2 \cdot \log_2\beta)^{-1}$.

Next, consider how the bound can be improved by appealing to Theorem 2. Since we can stochastically bound the absorption time sample paths of $\{V_k(i)\}$ by the sample paths of another chain $\{V'_k(i)\}$, with transition probability matrix

$$\mathbf{P}_{V'(i)}[j,k] = \begin{cases} q & k = j \\ p & k = 0 \end{cases} \tag{4.7}$$

for $2 \le j \le i$, and $\mathbf{P}_{V'(i)}[j,0] = 1$ for $j = 0, 1$. Theorem 2 guarantees that $E[T_n^V] \le E[T_n^{V'}]$. Recognizing that $P_{V'(i)}$ possesses principal submatrices which are strictly diagonal, we resort to (1.1) to obtain

$$E[T_i^{V'(i)}] = p^{-1} \quad \text{for} \quad 1 \le i \le n, \tag{4.8}$$

so that

$$E[T_n^V] \le E[T_n^{V'}] = n \cdot p^{-1} = O(n) \tag{4.9}$$

where the $O(n)$ complexity on expected run time holds because p is independent of n. It should be clear that since every element must be considered for membership in S, $E[T_n^V] = \Omega(n)$.

If the random graph G is generated asymmetrically, so that node j is made adjacent to node i with probability $p_{i,j} = p_{j,i} = 1 - q_{i,j} > 0$, then setting $q = \max \{ q_{i,j} \mid 1 \le j < i \}$ and $\beta = \frac{1}{q}$, the bound (4.6) is still valid. Further, the refined bound in (4.9) also holds, with $p = \min \{ p_{i,j} \mid i < j \le n, 1 \le i \le n-1 \}$. The complexity of the lower bound on expected run time in this case remains at $\Omega(n)$.

4.C Sequential Searches

We associate a Markov chain $\{W_k ; k \ge 0\}$ with the search algorithm described in 2.C as follows. Assume that our search begins at the right end of the list and set $W_0 = n$. If key K resides in the n^{th} position of the table (the probability of this is p_n) then the chain moves to state 0 (gets absorbed) in one step. Otherwise the chain moves to state $(n-1)$ and the search continues until the table is exhausted. We obtain a chain whose transition probability matrix is similar to (4.4), i.e.,

$$W_{k+1} = \begin{cases} 0 & \text{with probability } p_{n-k} \\ W_k - 1 & \text{with probability } q_{n-k} \end{cases} \tag{4.10}$$

where $q_{k+1} = 1 - p_{k+1}$ for $0 \le k \le n-1$. Observing that the hypothesis of Theorem 1 is satisfied, we set $q = \max \{q_1, \ldots, q_n\}$, and $\beta = q^{-1}$ to obtain

$$E[T_n^W] \le \log_\beta n + 0(1). \tag{4.11}$$

Indeed, using the stochastic bounding argument of the previous example, it can be shown that

$$E[T_n^W] < p^{-1} \tag{4.12}$$

where $p = \min \{p_1, p_2, \ldots, p_n\}$.

Consider how the same technique may be applied to a general search strategy (Knuth [3]). Let $p = 1 - q > 0$, and assume we are given a table of n numbers in ascending order. The algorithm compares key k with the $(p\,n)^{th}$ key and then iterates this procedure on smaller blocks. Let $C(n)$ denote the average number of comparisons required for a successful search on n records. Then it can be shown (see Knuth [3]) that

$$C(n) = 1 + p\,C(p\,n) + q\,C(q\,n) \tag{4.13}$$

for $n > 1$ and $C(1) = 0$. The binary search ($p = q = 0.5$) and the Fibonacci search ($p = \phi^{-1}$, $q = \phi^{-2}$, $\phi = 1.618$) are special cases of this search algorithm.

In order to demonstrate an asymptotic bound for $C(n)$, associate a Markov chain $\{W_k\}$ with the algorithm. Define $W_0 = n$ and

$$W_{k+1} = \begin{cases} \lceil p\,n \rceil & \text{with probability } p \\ \lceil q\,n \rceil & \text{with probability } q. \end{cases} \tag{4.13}$$

so that $\{W_k\}$ is a chain on the nonnegative integers. In a single step the chain $\{W_k\}$ moves from state n to either state $\lceil p\,n \rceil$ or state $\lceil q\,n \rceil$. There is a probability p that the search (approximately) reduces to a $(p\,n)$ element search and a probability q that it reduces to a $(q\,n)$ element search, after the first comparison. For large n, we may ignore the effect caused by non-integral values of $p\,n$ and $q\,n$.

Without loss of generality, we take $p > q$, so $\{W_k\}$ moves over states $\{0,1,2,3,\ldots,\lceil qn \rceil,\ldots,\lceil pn \rceil,\ldots,n\}$. Clearly,

$$E(W_1 \mid W_0 = n) = p \cdot \lceil pn \rceil + q \cdot \lceil qn \rceil < n \tag{4.14}$$

so that the hypothesis of Theorem 1 is satisfied, for $\beta = (p \cdot \lceil pn \rceil + q \cdot \lceil qn \rceil)^{-1}$. Thus, we obtain the average run time bound

$$E(T_n^W) \le \log_\beta n + O(1). \tag{4.15}$$

4.D Random sorts

Let $\{X_k; k \geq 0\}$ be a stochastic process representing the behavior of the random sort described in 2.D. Assume that we begin with a list that has n flags so that $X_0 = n$, and let X_k denote the number of flags in list $L^{(k)}$, $k \geq 1$. Observe that $\{X_k\}$ is *not* a Markov chain, since the values of the elements in each list $L^{(k)}$ determine the behavior of $\{X_k\}$, and the number of flags present is not sufficient to determine transitions. Further, it is easy to see that

$$E(X_{k+1} \mid X_k) \leq X_k + 1 \tag{4.16}$$

$\forall\, k \geq 1$, which is a weaker expectation condition than the one we require. Thus, we cannot resort to Theorem 1 directly. What (4.16) says is that the number of flags in $L^{(k+1)}$ is, on the average, at *most* one greater than the number of flags in $L^{(k)}$. To get around this, if we can guarantee that the number of flags in $L^{(k+m)}$ is, on the average, *less* than the number of flags in $L^{(k)}$, for some finite m, we may still use Theorem 1 in a slightly modified form.

Given $X_0 = r$, it is easy to demonstrate that

$$E(X_m \mid X_0 = r) \leq \frac{r}{\beta} \tag{4.17}$$

for any r, $1 \leq r \leq n$, and some value of m, $1 \leq m \leq n$. That is, even though X_1, X_2, etc. may exceed r, the number of flags must eventually fall *below* r within *at most* m steps, for some value of m, $1 \leq m \leq n$. In the worst possible scenario $m \sim O(X_k)$, for list $L^{(k)}$, so that all flagged elements in a particular list must be examined/selected in order to reduce the number of flags by one or more. On the average, however, this number will be less than X_k. Clearly, (4.17) generates the recurrence

$$E(X_{mj} \mid X_0 = n) \leq \frac{n}{\beta^j} \tag{4.18}$$

for $j \geq 1$, and some m, $1 \leq m \leq n$. Introduce a "sparser" process $\{Y_k\}$ defined at every m^{th} epoch of the process $\{X_k\}$ with $Y_0 = X_0 = n$ and $Y_j = X_{mj}$ for $j \geq 1$ and m fixed. Then (4.18) yields

$$E(Y_j) \leq \frac{n}{\beta^j} \tag{4.19}$$

and finally, by Theorem 1,

$$E(T_n^Y) \leq \log_\beta n + \frac{\beta}{\beta - 1} \tag{4.20}$$

for some $\beta > 1$. Moving from process $\{Y_k\}$ to process $\{X_k\}$, we see that, since $1 \leq m \leq n$,

$$E(T_n^X) \leq n \log_\beta n + O(n) - O(n \log n) \tag{4.21}$$

The random variable T_n^X does not include the amount of work required, at each step k of the algorithm, to determine the number of flags present in $L^{(k)}$. Since this work is $O(n)$, we obtain the average run time of the random sorting algorithm on a uniprocessor to be $O(n^2 \log n)$. In the case of an n processor sort, if we agree that processors execute asynchronously and only one processor executes at a time, then the previous $O(n)$ work required to determine the number of flags present in the list at each step can be ignored. Thus, the n processor distributed sort yields an average run time that is $O(n \log n)$.

In order to determine a lower bound on uniprocessor sorting time, we proceed as follows. Define $\{X_k'; k \geq 0\}$ to be an n-urn game with parameter $q = \frac{n-1}{n}$. The process $\{X_k\}$ moves from a state with r flags to a state with at least $(r-1)$ flags in a minimum of one step. Since movement from an r flag state to an $(r-1)$ flag state can occur with probability 1 for process $\{X_k\}$, we must ensure (if a lower bound is to be had) that process $\{X_k'\}$ moves at least as fast. Choosing $q = \frac{n-1}{n}$ for the urn game process achieves precisely this. Theorem 2 yields

$$E(T_n^X) \geq E(T_n^{X'}) = \frac{\log n}{\log q^{-1}} + 0(1) \tag{4.22}$$

Thus, the uniprocessor random sort yields an average run time lower bound of $\Omega(n \log n)$. Repeating the argument used in the case of the upper bound, the n processor distributed sort yields a lower bound of $\Omega(\log n)$.

4.E Dependent sevice queuing systems

In this example, we seek a bound on a passage time, i.e., the mean time for this general queueing system to drift from an n customer queue to an empty queue. Using Corollary 1.1 (see example (v)), an upper bound can be obtained provided the rows of Q satisfy a necessary condition. If

$$E(X_1 \mid X_0 = n) \leq \frac{n}{\beta} \tag{4.23}$$

$\forall\, n \geq 1$ and $\beta > 1$, then since the grouping of states in Q is "natural", Theorem 1 yields

$$E(T_n^X) \leq \log_\beta n + \frac{\beta}{\beta - 1} \tag{4.24}$$

regardless of the size of the groups. Consider a special case of this system, namely, the $M \mid GI \mid 1$ queue, where arrivals occur singly, customers are of only one type, service times are independent, and group size is 1. For the $M \mid GI \mid 1$ queue,

$$E(X_1 \mid X_0 = n) = n - 1 + \rho \leq \frac{n}{\beta} \tag{4.25}$$

for all n, and $1 < \beta \leq \dfrac{n}{n + \rho - 1}$. Applying Theorem 1, we obtain

$$E(T_n^X) \leq \log_\beta n + O(1). \tag{4.26}$$

It is simple to obtain a lower bound on $E(T_n^X)$ for the $M/GI/1$ queue. Construct an urn game process $\{Y_k\}$ with parameters (n, q) and observe that for virtually any value of q, $T_n^X \geq_{st} T_n^Y$. Using (3.16) it follows that

$$E(T_n^X) \geq E(T_n^Y) - \Omega(\log n) \tag{4.27}$$

REFERENCES

[1] R. Kemp, *Fundamentals of the Average Case Analysis of Particular Algorithms*, Wiley-Teubner, 1984.

[2] M. Hofri, *Probabilistic Analysis of Algorithms*, Springer-Verlag, 1987.

[3] D. Knuth, *The Art of Computer Programming*, Vol. 3, *Sorting and Searching*, Addison-Wesley, 1973.

[4] M. Neuts, "Some Explicit Formulas for the Steady-State Behavior of the Queue with Semi-Markovian Service Times," *Adv. Appl. Prob.*, 9, pp. 141–157, 1977.

[5] V. Rego and W. Szpankowski, "On the Quality of Approximations for High Speed Rings," *Proceedings of the International Symposium on High Performance Computer Systems*,, Ed. E. Gelenbe, North-Holland, pp. 179–193, 1988.

[6] O.N. Stavskaya and I.I. Pyatetskii-Shapiro, "On certain properties of homogeneous nets of spontaneously active elements," *Problemi Ciberneriki*, 20, M. Nauka, pp. 91–106, 1968.

[7] W. Szpankowski and V. Rego, "Yet Another Application of a Binomial Recurrence: Order Statistics," *working paper*, Purdue CSD-TR-765, April 1988.