A PRESORTEDNESS METRIC FOR ENSEMBLES OF DATA SEQUENCES

R.S. Valiveti and B.J. Oommen

SCS-TR-174, MAY 1990

School of Computer Science, Carleton University
Ottawa, Canada, KIS 5B6

A Presortedness Metric for Ensembles of Data Sequences

R.S. Valiveti and B.J. Oommen *
School Of Computer Science
Carleton University
Ottawa, Canada, K1S 5B6

Abstract

Sorting is a well studied problem in the field of computer science. Many excellent sorting methods are known today, including the recent adaptive methods which sort all sequences of data (or permutations) and perform particularly well on sequences that are "almost sorted" according to some "presortedness" measure. The methods existing in the literature have focussed on assigning a "presortedness" measure to a **single** input sequence. The particular measure in question can then be related to the computational complexity of the algorithm being utilized. Such a measure, in itself, cannot be used to predict the expected complexity of an algorithm. A comparison of any two sorting algorithms (in the expected sense) necessarily requires a knowledge of the distribution of the data sequences presented to the sorting program.

In an attempt to capture the nature of the distribution of data sequences presented to a sorting program, we propose a presortedness metric which characterizes the properties of an **ensemble** of data sequences by a single parameter. This parameter indicates how sorted or unsorted the ensemble of random data sequences is. Various theoretical properties of the metric are derived. The estimation procedure for determining the Maximum Likelihood Estimate of the "Ensemble Presortedness metric" for a given ensemble of data sequences is also included. The results have been experimentally validated.

Keywords:
Sorting Algorithms, Expected Time Complexity, Probability distribution, Parametric approximation, Parameter estimation.

*Both authors partially supported by Natural Sciences and Engineering Research Council Of Canada.

1 Introduction

Sorting is a problem of paramount importance in the field of computer science. The optimal sequential time complexity of sorting a sequence of M elements is $O(M \log M)$. While many optimal techniques are known, there also exist algorithms (e.g. Quicksort) which are not optimal in the worst case sense. These algorithms can be presented with specific sequence of data elements, for which a running time of $O(M^2)$ results. This phenomenon has led researchers to seek alternate sorting methods which never encounter this worst case quadratic behaviour. The key idea motivating this research is to find methods whose time complexity increases "smoothly" as a function of some measure of "presortedness" of the input sequence [1, 5, 6].

The class of "adaptive" sorting methods achieve exactly this goal. These methods monitor some measure of the input sequence, such as the number of inversions [2] or the number of oscillations [3]. Such a measure in essence defines the "distance" of the input data sequence, from its sorted form. Algorithms based on these measures have a time complexity which is $O(M)$ when the "distance" measure is small and approaches the lower bound for sorting as the measure increases [4].

While such adaptive algorithms perform rather well on specific input sequences, an objective comparison of these algorithms with other existing methods (or even among themselves) cannot be achieved by just studying the "distance metrics" of individual sequences. An example will help clarify this point. Let us suppose that the set of data elements to be sorted is $\{1, 2, 3, 4, 5\}$. If an Insertion Sort (IS) is presented with the sequence $\pi_1 = \langle 1, 2, 3, 5, 4 \rangle$ the sorting is achieved by performing exactly one swap and $O(M)$ comparisons. However if the input sequence was $\pi_2 = \langle 5, 4, 3, 2, 1 \rangle$, $O(M^2)$ swaps would be necessary.

As opposed to the Insertion Sort, let us suppose that the same data sequences were presented to an adaptive scheme, such as the Adaptive HeapSort (AHS) described in [3]. The AHS would take $O(M)$ operations to sort each of the sequences π_1 and π_2. The question now before us is to objectively compare IS and AHS.

If the data to be sorted was only of the type π_1, clearly the IS is superior. IS only make $O(M)$ comparisons and also avoids the overhead encountered in maintaining and updating the heap used by the AHS. However, if the data set was always of the type π_2, clearly the AHS is superior. In order to objectively compare the two algorithms, not only should the presortedness metric of a sequence be considered, but the comparison must also involve the probability distribution on the set of permutations on the data set.

Thus our stand in this paper is that whereas distance metrics such as Inversions [2], Runs, and especially Oscillations [3] are extremely powerful in quantifying the presortedness of a **single** data sequence, they are incapable of characterizing the presortedness of **ensembles** of sequences. Instead of assigning metrics to each data sequence contained in an ensemble, we assign a single Ensemble Presortedness measure to the ensemble as a whole. This single parameter reflects the divergence of the various sequences in this ensemble from the purely sorted from. This is the fundamental contribution of this paper.

Strictly speaking, given a set of M elements, the probability associated with each possible data sequence can be fully defined by enumerating $M!$ probabilities. Apart from this characterization being unrealistic and impractical, it does not provide a single parameter with which we can compare two ensembles of data sequences. In this light, as alluded to above, we shall present a new metric called the Ensemble Presortedness metric. This metric is completely specified by a single parameter ρ. Given the parameter ρ, the Distribution on all possible data sequences is completely described by means of an $M-$dimensional probability vector. The i^{th} component of this vector is recursively defined in terms of the $(i-1)^{st}$, and is obtained by merely multiplying the latter by ρ. Indeed if $\rho = 0$, it represents an ensemble consisting only of the sorted sequence. Furthermore we shall show that as ρ increases, the number of inversions increases.

The approach taken in this paper is as follows. We shall first show how to generate ensembles of data sequences, given the parameter ρ. Subsequently we also study the problem of evaluating the parameter ρ for a given ensemble. In other words, given an ensemble of data sequences, we present an estimation procedure by which the Maximum Likelihood Estimate (MLE) of the parameter ρ can be obtained. A natural consequence is that we can now compare sorting algorithms not only in terms of their worst case time complexity but also their expected time complexity.

Section 2, presents our model of data sequence generation. This model is based on a fairly general model for permutation generation, called the S-model. Several properties of the ρ-model of data sequence generation are derived in the same section. Section 3 describes the procedures used to obtain the MLE of ρ. Section 4 presents our experimental results.

2 The S-model for Data Sequence Generation

Let $\mathcal{R}$ be the set of records $\{R_1, R_2, \ldots, R_M\}$, where there is an implicit ordering among the elements of $\mathcal{R}$ that $R_i < R_j$ if $i < j$. In order to extract features from an ensemble of data sequences obtained from the set $\mathcal{R}$, a very simple method would recommend maintaining the $M!$ frequency counters which estimate the probabilities with which the given source generates the various possible data sequences (or permutations). Clearly such a strategy is infeasible even when M is as small as 10. Considering the above, our first major deviation is to model permutation generators in a fashion which requires a linear (as opposed to $M!$) number of parameters. This model, called the S-model, was initially proposed by Oommen and Ng in [7].

In an S-model, we assume that the underlying data generation strategy can be completely specified by a control vector of dimension M. This vector $\mathcal{S}$ is an $M \times 1$ probability vector denoted by $[s_1, s_2, \ldots, s_M]^T$ which satisfies:

$$\sum_{i=1}^{M} s_i = 1.$$

Let $\mathcal{V}$ be any subset of $\mathcal{R}$. We define the vector $\mathcal{S}_{\mathcal{V}}$ as the **conditional** control vector of (normalized probabilities) in which only the quantities corresponding to the members of $\mathcal{V}$ have non-zero values. Thus $\mathcal{S}_{\mathcal{V}}$ consists of normalized probabilities $[s_i']$, where:

$$s_i' = \begin{cases} 0 & \text{if } R_i \notin \mathcal{V} \\ \dfrac{s_i}{\sum_{R_i \in \mathcal{V}} s_i} & \text{otherwise} \end{cases} \tag{1}$$

Observe that the vector $\mathcal{S}$ is exactly equivalent to $\mathcal{S}_{\mathcal{R}}$.

Given the vector parameter $\mathcal{S}$, we assume that the underlying generation strategy is as follows. Let Π be the set of all permutations of the elements of $\mathcal{R}$. For the sake of explanation, let $X = \langle x_1, x_2, \ldots, x_M \rangle$ be a randomly generated data sequence ($X \in \Pi$). The generation of a random data sequence X clearly consists of randomly assigning a position to each element of $\mathcal{R}$. The S-model proposes that this is done by computing the prefix subsequence of X in succession. Initially x_1 is randomly assigned an element in $\mathcal{R}$, based on the control vector $\mathcal{S}$(which is the same as $\mathcal{S}_{\mathcal{R}}$). By this we mean that element R_i is chosen with probability s_i. The subsequent problem is simply one of generating a (random) sequence of the $(M-1)$ elements of $\mathcal{R} - \{x_1\}$. The S-model proposes that this generation is done using the conditional probability vector $\mathcal{S}_{\mathcal{V}}$ and the process is recursively repeated by successively updating $\mathcal{V}$. The algorithmic form of this procedure (referred to as Algorithm S-model) is presented in Program 1.

Having defined the S-model for data sequence generation, the next question is that of specifying the individual components of the vector $\mathcal{S}$. To enable us to have a single parameter distribution metric to compare the presortedness of ensembles, we define s_i as follows:

$$s_i = k\rho^{i-1}$$

Observe that this implies that the complete vector $\mathcal{S}$ is given by:

```
Algorithm S-model

Input:
    The control vector $\mathcal{S} = [s_1, s_2, \ldots, s_M]^T$,
    where the components of $\mathcal{S}$ satisfy $\sum_{i=1}^{M} s_i = 1$.
Output:
    A permutation $\langle x_1, x_2, \ldots, x_M \rangle$ of $\mathcal{R}$.
Method:
    begin
        $\mathcal{V} := \mathcal{R}$;
        for i := 1 to M do
        begin
            Compute the conditional distribution $\mathcal{S}_{\mathcal{V}}$
            according to (1).
            Choose an element $x_i$ according to this distribution.
            $\mathcal{V} := \mathcal{V} - \{x_i\}$
        endfor
    end

End Algorithm S-model
```

Program 1: S-model for permutation generation

$$\mathcal{S} = k[1, \rho, \rho^2, \ldots, \rho^{M-1}]^T. \tag{2}$$

where k is a normalizing factor chosen so as to render $\mathcal{S}$ to be a probability vector. It will be seen that if $\rho < 1$, the components of $\mathcal{S}$ in (2) are in the descending order of magnitude. This fact can be used to show that the data sequence $\langle R_1, R_2, \ldots, R_M \rangle$ occurs with the highest probability. In the limit as $\rho \to 0$, the data sequence $\langle R_1, R_2, \ldots, R_M \rangle$ appears with probability 1. In other words, the ensemble generated by the model consists only of the sorted list.

On the other hand, if $\rho > 1$, the components of $\mathcal{S}$ are in ascending order. As the value of ρ becomes increasing larger, the last component of $\mathcal{S}$ becomes much larger than the rest, thereby implying that the data sequence $\langle R_M, R_{M-1}, \ldots, R_1 \rangle$ will occur with the highest probability. In the limit as $\rho \to \infty$, this probability reaches the limiting value of unity and in this case the only sequence presented to the sorting algorithm is the reverse of the sorted list. It will be shown in Section 2.1 that the expected number of inversions increases monotonically with ρ.

Since intermediate value of ρ are of interest, we observe that the value $\rho = 1$ corresponds to the case in which the components of $\mathcal{S}$ are all equal. In this case all the $M!$ data sequences are generated by the ρ-model with equal probability. This corresponds to the uniformity assumption traditionally made in expected case analysis.

2.1 Properties of the S-model

We use the notation $R_u \to j$ to denote the fact that the element R_u appears at the j^{th} position in the permutation. Furthermore, $R_u, R_v \to < i, j >$ represents the fact that the elements R_u and R_v appear at the positions i and j in the permutation, respectively. The extensions to this notation, where we are specifying the position of more than two elements are straightforward and are interpreted in an intuitive fashion.

$R_u \prec R_v$ denotes the event that element R_u preceds element R_v in a given permutation.

We now present some of the theoretical properties of the data sequences generated by the S-model. The first result derived in the context of the general S-model [7] is given below for the case when the probability vector is described in terms of the Ensemble Presortedness Metric ρ.

Lemma 1 *Let $\mathcal{S} = [1, \rho, \rho^2, \ldots, \rho^{M-1}]$ be the (M-dimensional) control vector of the S-model. Also let u, v be two distinct indices belonging to the set $\{1, 2, \ldots, M\}$. Then, in the ensemble of data sequences generated, the probability that a record R_u preceeds another record R_v is given by:*

$$b(u, v) = Pr(R_u \prec R_v) = \frac{\rho^u}{\rho^u + \rho^v}.$$

Proof:

For ease of derivation, we let $\mathcal{S} = k[1, \rho, \rho^2, \ldots, \rho^{M-1}]^T = [s_1, s_2, \ldots, s_M]^T$.. With this notation, s_u (s_v) is the probability that record R_u (R_v) is accessed.

Let $\xi_{u,v}(n)$ be the event that position n contains the first appearance of R_u or R_v. That is, neither of them have appeared before and that the n^{th} position contains one of them. Note that the events in the set $\{\xi_{u,v}(n)$ for $n = 1, 2, \ldots, M-1\}$ are mutually exclusive and collectively exhaustive. By virtue of the generation scheme, R_u ultimately precedes R_v if it is selected before record R_v.

We first consider the case when $n = 1$. Given $\xi_{u,v}(1)$, R_u ultimately precedes R_v in the data sequence every time R_u is the **first** element selected. Clearly, because of the strategy used in Algorithm S-model, the probability that R_u appears in the first position is s_u. Similarly, the probability that R_v ultimately precedes R_u in the data sequence is s_v. Thus,

$$Pr(R_u \prec R_v | \xi_{u,v}(1)) = \frac{s_u}{s_u + s_v}.$$

We now consider the general case in which the value of the parameter n takes on values other than 1. Since R_u and R_v have so far been not selected, and $n-1$ other elements have been already included in the data sequence, the set $\mathcal{V}$ in Algorithm S-model must be of cardinality $M-n+1$. Moreover $\mathcal{V} \supseteq \{R_u, R_v\}$. The generation of the element for position n will be done using the conditional probability vector $\mathcal{S}|\mathcal{V} = [s'_1, s'_2, \ldots, s'_M]^T$. CLearly s'_u and s'_v will be non-zero, since both s_u and s_v are positive. The conditional probability that R_u precedes R_v given $\xi_{u,v}(n)$ is simply given by:

$$\begin{aligned} Pr(R_u \prec R_v|\xi_{u,v}(n)) &= Pr(R_u \text{ is selected for position } n|\xi_{u,v}(n)) \\ &= Pr(R_u \text{ is selected for position } n)/Pr(\xi_{u,v}(n)) \\ &= \frac{s'_u}{s'_u + s'_v} \\ &= \frac{s_u}{s_u + s_v} (\text{since } s'_u \text{ and } s'_v \text{ are scaled by the same factor}) \end{aligned}$$

Now by the laws of total probability, the required probability can be written down as:

$$\begin{aligned} b(u,v) &= \sum_{n=1}^{M-1} Pr(R_u \prec R_v|\xi_{u,v}(n)) Pr(\xi_{u,v}(n)) \\ &= Pr(R_u \prec R_v|\xi_{u,v}(n)) \sum_{n=1}^{M-1} Pr(\xi_{u,v}(n)) (\text{since } Pr(R_u \prec R_v|\xi_{u,v}(n)) \text{ is indep. of } n) \\ &= Pr(R_u \prec R_v|\xi_{u,v}(n)) \\ &= \frac{s_u}{s_u + s_v} = \frac{\rho^u}{\rho^u + \rho^v}. \end{aligned}$$

and the Lemma is proved. □

Theorem 1 *Let $\mathcal{S} = [1, \rho, \rho^2, \ldots, \rho^{M-1}]$ be the (M-dimensional) control vector of the S-model. Then, in the ensemble of data sequences generated, the expected number of inversions is given by:*

$$E[\text{Inversions}|\rho] = \sum_{j>i} \frac{\rho^{j-i}}{1+\rho^{j-i}}.$$

Proof:

Let Y_{ij} be the random variable defined as follows:

$$Y_{ij} = \begin{cases} 1 & \text{if Record } R_i \text{ appears before record } R_j \text{ in a data sequence} \\ 0 & \text{otherwise} \end{cases}$$

Then the number of inversions in a data sequence X is given by:

$$\text{Inversions} = \sum_{j>i} Y_{ji}$$

Hence the expected number of inversions in the data sequences characterized by ρ is:

$$E[\text{Inversions}|\rho] = \sum_{X \in \Pi} \sum_{j>i} Y_{ji} Pr(X). \tag{3}$$

We observe that $\sum_X Y_{ji} Pr(X)$ is very simply the probability, $b(j,i)$ that record R_j precedes R_i in a random data sequence. With this understanding, (3) can be rewritten as:

$$E[\text{Inversions}|\rho] = \sum_{j>i} b(j,i). \tag{4}$$

Using Lemma 1, $b(j,i)$ can be written down as:

$$\begin{aligned} b(j,i) &= \frac{\rho^{j-1}}{\rho^{j-1} + \rho^{i-1}} \\ &= \frac{\rho^{j-i}}{\rho^{j-i} + 1}. \end{aligned} \tag{5}$$

Substituting (5) in (4), we obtain the desired result. □

Observe from (5) that $b(j,i)$ (for all $j > i$) increases as ρ increases. As a natural consequence, the expected number of inversions also increases. Moreover when $\rho = 0$, $b(i,j) = 0$ whenever $j > i$. Also for $j > i$, $\lim_{\rho \to \infty} b(j,i) = 1$. This implies that $\lim_{\rho \to \infty} E[\text{Inversions}|\rho] = \sum_{j>i} 1 = M * (M-1)/2$. This is obviously the maximum number of possible inversions in a data sequence of length M. We are currently investigating the relationship between the parameter ρ and the expected values of other "presortedness" metrics such as Runs and Oscillations.

Besides (5), a more general probabilistic property of the data sequences can be derived [8]. In the case when the vector $\mathcal{S}$ is specified in terms of the parameter ρ, the following result is true.

Theorem 2 *Let $\mathcal{S} = [1, \rho, \rho^2, \ldots, \rho^{M-1}]$ be the (M-dimensional) parameter vector of the S-model. Also let $R_u, R_{v_1}, \ldots, R_{v_K}$ be distinct elements of $\mathcal{R}$. Then the probability that the element R_u precedes each one of the elements of the set $\{R_{v_1}, \ldots, R_{v_K}\}$ in any data sequence generated by Algorithm S-model, is given by:*

$$Pr(R_u \prec \{R_{v_1}, \ldots, R_{v_K}\} \mid \mathcal{S}) = \frac{\rho^u}{\rho^u + \sum_{j=1}^{K} \rho^{v_j}} \tag{6}$$

Proof:

For ease of derivation, we let $\mathcal{S} = k[1, \rho, \rho^2, \ldots, \rho^{M-1}]^T = [s_1, s_2, \ldots, s_M]^T$.. We prove this result by induction. Observe that the result stated in Theorem 2 must be proved for all (feasible) values of K and M. Since $\{R_{v_1}, \ldots, R_{v_K}\}$ can at most include all the elements of

$\mathcal{R} - \{R_u\}$, it follows that $K \leq M - 1$. A traditional induction on K would merely involve proving that the result is true for all K satisfying $K \leq M - 1$. Although M is a constant in our original setting, it must be treated as a parameter, to render the proof valid. The rationale for considering M as a parameter is as follows. For a given M the result could be proved by induction, without thereby implying that the result would continue to be true if the dimensionality of the problem M is changed. Observe that (6) states the assertion of the theorem independent of the dimension of the set $\mathcal{R}$.

As a consequence of the above reasoning, it appears that a "two-dimensional induction", i.e. an induction on two variables is unavoidable. However, we shall reduce this into a single parameter induction. This is done as follows. For a given M, we shall prove that (6) is valid if it is valid for all values of K and $M = M - 1$. The formal steps of the proof follow.

The theorem is trivially valid if $K = M - 1$. In this case, the denominator of (6) essentially consists of the sum $\sum_{i=1}^{M} s_i$, which is unity. The expression in (6) thus simplifies to s_u — which is indeed the probability that the element R_u appears before each of the other elements of $\mathcal{R}$, i.e. in the first position.

We now proceed by induction on the parameter M. Let us assume that the theorem holds for all values of $K \leq M - 1$, for a specific value M_0 of the parameter M. We will establish that the same property holds for $M = M_0 + 1$ as well.

To establish the basis for the induction, we examine the case when $M = 2$. In this case the only meaningful value of K is 1. Since $K = M - 1$, the result indeed holds for the case when $M = 2$. We now proceed with the induction step.

Let $\mathcal{S} = [s_1, s_2, \ldots, s_M]^T$, where $M = M_0 + 1$. In order to derive the probability that the element R_u preceds each one of the elements in $\{R_{v_1}, \ldots, R_{v_K}\}$, we note that this event happens if:

(i) R_u appears in the first position.

(ii) Any element not contained in the set $\{R_u\} \cup \{R_{v_1}, \ldots, R_{v_K}\}$ appears in the first position and *then* R_u (recursively) preceds $\{R_{v_1}, \ldots, R_{v_K}\}$ in the permutation of the remaining $M - 1$ elements.

Note that the events labelled as (i) and (ii) are mutually exclusive and hence using the notation defined in this section, from the law of total probability, we have:

$$\begin{aligned} &Pr(R_u \prec \{R_{v_1}, \ldots, R_{v_K}\} \mid \mathcal{S}) \\ &\quad = Pr(R_u \rightarrow 1) + \sum_{R_l \notin \{R_u\} \cup \{R_{v_1}, \ldots, R_{v_K}\}} Pr(R_l \rightarrow 1) Pr(R_u \prec \{R_{v_1}, \ldots, R_{v_K}\} \mid \mathcal{S}_{\mathcal{R} - \{R_l\}}) \end{aligned}$$

$$= \; s_u + \sum_{R_l \notin \{R_u\} \cup \{R_{v_1}, \ldots, R_{v_K}\}} s_l Pr(R_u \prec \{R_{v_1}, \ldots, R_{v_K}\} \mid \mathcal{S}_{\mathcal{R}-\{R_l\}}) \tag{7}$$

We note that $\mathcal{S}_{\mathcal{R}-\{R_l\}}$ is a control vector of dimension M_0, and can be denoted as $[s'_\beta \mid \beta \in \{1, 2, \ldots, M_0\} - \{l\}]^T$ and hence each of the probabilities in the RHS of equation (7), can be calculated using the induction hypothesis. To illustrate, let us consider the term:

$$Pr(R_u \prec \{R_{v_1}, \ldots, R_{v_K}\} \mid \mathcal{S}_{\mathcal{R}-\{R_l\}}) = \frac{s'_u}{s'_u + \sum_{j=1}^{K} s'_{v_j}}. \tag{8}$$

Notice that $s'_\beta = s_\beta/(1 - s_l)$, since $1 - s_l$ is the normalizing factor for **all** the probabilities in the control vector $\mathcal{S}_{\mathcal{R}-\{R_l\}}$. Therefore:

$$Pr(R_u \prec \{R_{v_1}, \ldots, R_{v_K}\} \mid \mathcal{S}_{\mathcal{R}-\{R_l\}}) = \frac{s_u}{s_u + \sum_{j=1}^{K} s_{v_j}}. \tag{9}$$

Notice that this probability does not dependent on the index l. Combining (9) with (7), we have:

$$\begin{aligned}
Pr(R_i \prec \{R_{v_1}, \ldots, R_{v_K}\} \mid \mathcal{S}) &= s_u + \sum_{R_l \notin \{R_u\} \cup \{R_{v_1}, \ldots, R_{v_K}\}} s_l \frac{s_u}{s_u + \sum_{j=1}^{K} s_{v_j}} \\
&= s_u \left(1 + \frac{\sum_{R_l \notin \{R_u\} \cup \{R_{v_1}, \ldots, R_{v_K}\}} s_l}{s_u + \sum_{j=1}^{K} s_{v_j}} \right) \\
&= \frac{s_u}{s_u + \sum_{j=1}^{K} s_{v_j}} \cdot \left(\sum_{l=1}^{M_0+1} s_l \right) \\
&= \frac{s_u}{s_u + \sum_{j=1}^{K} s_{v_j}}.
\end{aligned} \tag{10}$$

Let $f(K, M)$ represent the fact that (6) is true for specific values of K and M. Thus, as a consequence of (10) we have,

$$\{\forall K \leq M - 1\} [f(K, M)] \Rightarrow \{\forall K \leq M - 1\} [f(K, M + 1)]$$

Since f(M, M+1) is trivially true, we have effectively proved that

$$\{\forall K \leq M - 1\} [f(K, M)] \Rightarrow \{\forall K \leq M\} [f(K, M + 1)]$$

This completes the proof. □

3 Evaluation of ρ, the Ensemble Presortedness Metric

Till now we have focussed on the problem of obtaining data sequences which are generated based on an M-dimensional vector $\mathcal{S}$. We demonstrated how the components of this vector $\mathcal{S}$ can be completely specified in terms of a single parameter ρ, called the Ensemble Presortedness metric. We derived several properties of the data sequences generated under this model. In this section, we concentrate on the aspect of estimating the best value of ρ which describes a given data sequence ensemble. We state with a mere sketch of proof, our first result in this regard.

Theorem 3 *The maximum likelihood estimate of the parameter ρ can be obtained only if we maintain $O(2^M)$ statistics.*

Proof:

The proof mainly involves writing the expression for the likelihood function and making some simplifications. It is shown in [8] that the information to be maintained essentially consists of the following counters:

1. the number of times record R_i appears in position j in the data sequence.

2. The number of times the record pair R_u, R_v appears in the first two positions etc.

A simple enumeration of these counters leads to the required result. □

Theorem 3 has established that in order to rigorously obtain the Maximum Likelihood Estimate for the parameter ρ, we are required to maintain $O(2^M)$ statistics. This is an impractical amount of information to maintain. For example, even if M is as small as 10, we would be faced with maintaining almost 1000 counters. This is clearly infeasible. The amount of information to be extracted from data sequences can be substantially reduced if we only chose to observe events of the form "Record R_u precedes record R_v". This implies the maintenance of $\binom{M}{2}$ counters. Our next result develops the method to be used for estimating the parameters, if only these $O(M^2)$ statistics are available.

Theorem 4 *Let ${}_uN_v$ represent the number of times the element R_u appears before the element R_v, in the ensemble of N data sequences that were observed. If only the frequency counts of the form ${}_uN_v$ are available for all pairs u, and v, then the maximum likelihood estimate for the parameter ρ can be obtained as the solution of the non-linear equation given below:*

$$\sum_{i<j} \frac{j-i}{1+\rho^{j-i}} = \sum_{i<j} \frac{{}_iN_j}{N}(j-i). \tag{11}$$

Proof:

We emphasize that in this part of the proof, we are not attempting to maximize the likelihood of generating the N sample data sequences presented to us. Instead we focus on maximizing the likelihood of explaining the frequencies of the events of the form "element R_i precedes element R_j" (symbolically represented as "$R_i \prec R_j$") given only the $\binom{M}{2}$ statistics. For ease of derivation, we let $\mathcal{S} = k[1, \rho, \rho^2, \ldots, \rho^{M-1}]^T = [s_1, s_2, \ldots, s_M]^T$.

Consider the event in which the element R_i appeared before the element R_j for a total of ${}_iN_j$ times. Since the N sample permutations are statistically independent, we can write down the probability of occurrence of this event as:

$$Pr(R_i \prec R_j \text{ occurs } {}_iN_j \quad \text{times}) = \binom{N}{{}_iN_j} [Pr(R_i \prec R_j)]^{{}_iN_j} [Pr(R_j \prec R_i)]^{{}_jN_i}$$

Note that as a corollary of Theorem 2, we have the following result:

$$\begin{aligned} Pr(R_i \prec R_j) &= \frac{s_i}{s_i + s_j} \\ Pr(R_j \prec R_i) &= \frac{s_j}{s_i + s_j} \end{aligned}$$

and hence,

$$\begin{aligned} Pr(R_i \prec R_j \,\text{occurs}\, {}_iN_j \quad \text{times}) &= \binom{N}{{}_iN_j} \left(\frac{s_i}{s_i + s_j}\right)^{{}_iN_j} \left(\frac{s_j}{s_i + s_j}\right)^{{}_jN_i} \\ &= \binom{N}{{}_iN_j} \frac{s_i^{{}_iN_j}\, s_j^{{}_jN_i}}{(s_i + s_j)^N} \qquad (12) \end{aligned}$$

The statistics extracted from the data samples consists of counts ${}_iN_j$, for all $i \neq j$, i.e. M(M-1)/2 counters and these are sufficient to evaluate terms of the form (12).

In order to arrive at the best estimates for the control vector $\mathcal{S}$, we have to maximize the quantity $Pr[\cap_{i<j}(R_i \prec R_j$ occurs ${}_iN_j$ times)]. Assuming that these events are *statistically independent* [1], we shall consider the likelihood function $\mathcal{L}$ (given only the statistics ${}_iN_j$) as shown below:

$$\begin{aligned} \mathcal{L} &\triangleq Pr[\cap_{i<j}(R_i \prec R_j \text{ occurs } {}_iN_j\text{times}] \\ &= \prod_{i<j} Pr[(R_i \prec R_j \text{ occurs } {}_iN_j\text{times}] \\ &= \prod_{i<j} \binom{N}{{}_iN_j} \frac{s_i^{{}_iN_j}\, s_j^{{}_jN_i}}{(s_i + s_j)^N} \qquad \text{from (12)} \end{aligned}$$

[1] This assumption is strictly not true. See [8] for a counter example.

Hence the logarithm of the likelihood function, $\ell \triangleq \log \mathcal{L}$, can be written as:

$$\begin{aligned} \ell &= Z + \sum_{i<j} \{{}_iN_j \log s_i + {}_jN_i \log s_j - N \log(s_i + s_j)\} \\ &\qquad\qquad (\text{substituting } s_i = k\rho^{i-1}) \\ &= Z + \sum_{i<j} \left\{{}_iN_j(i-1)\log k\rho + {}_jN_i(j-1)\log k\rho - N\log(k\rho^{i-1} + k\rho^{j-1})\right\} \qquad (13) \end{aligned}$$

where Z stands for the quantity $\sum_{i<j} \log \binom{N}{{}_iN_j}$.

It is clear that the MLE for ρ can simply be found by differentiating (13) w.r.t. the parameter ρ and setting the result to 0. Hence we obtain,

$$\textstyle\sum_{i<j} \quad \frac{{}_iN_j}{\rho}(i-1) + \frac{{}_jN_i}{\rho}(j-1) - \frac{N}{(k\rho^{i-1} + k\rho^{j-1})}\left[(i-1)k\rho^{i-2} + (j-1)k\rho^{j-2}\right] = 0$$

or,

$$\textstyle\sum_{i<j} \quad \frac{1}{\rho}({}_iN_j i + {}_jN_i j - N) - \frac{N}{\rho + \rho^{j-i+1}}\left\{(i-1) + (j-1)\rho^{j-i}\right\} = 0$$

which simplifies to,

$$\textstyle\sum_{i<j} \quad \left\{\frac{{}_iN_j}{N} i + \frac{{}_jN_i}{N} j - 1\right\} - \frac{1}{1+\rho^{j-i}}\left\{(i-1) + (j-1)\rho^{j-i}\right\} = 0$$

or equivalently,

$$\textstyle\sum_{i<j} \quad \frac{j-i}{1+\rho^{j-i}} = \sum_{i<j} \frac{{}_iN_j}{N}(j-i).$$

and the theorem is proved. □

Remark: It must be noted that the above equation is a non-linear equation involving ρ. Clearly, an analytic solution is not feasible. In general it can be solved by using a root-solving technique such as the Newton-Raphson (NR) method.

4 Experimental Results

Having completely described the model for data sequence generation and given an estimation procedure to evaluate the Ensemble Presortedness metric (ρ), is not out of place for us to examine how effectively this quantity reflects the presortedness of various ensembles. To do this, various ensembles of sequences of data were generated. Each ensemble was characterized by the mean number of inversions. The effect of the mean number of inversions on the MLE of the parameter ρ was studied. The experiments were conducted as follows.

For a given number of elements M we allow at most $MAXINV = M * (M-1)/4$ inversions in any random data sequence examined. This number represents half of the maximum number of inversions possible in a list of size M. The ensemble characteristics were fixed by imposing a distribution $Q = [q_0, q_1, q_2, \ldots, q_{MAXINV}]^T$ on the number of inversions contained in any particular data sequence. In other words, q_j represents the probability of a random data sequence possessing exactly j inversions. For our experiments, we chose $\mathcal{Q}$ to obey:

$$\begin{aligned} \mathcal{Q} &= [q_0, q_1, q_2, \ldots, q_{MAXINV}]^T \\ &= A[1, (1-\delta), (1-\delta^2), \ldots, (1-\delta)^{MAXINV-1}]^T \end{aligned}$$

where δ is a parameter used to generalize the experiments, and A is a normalizing factor chosen so as to ensure that the components of $\mathcal{Q}$ sum to unity [2].

The method of random data sequence generation for each ensemble was as follows. For a specific value of M, various ensembles of data sequences were generated, based on values of δ in the interval $[-0.9, 0.9]$. Negative values of δ imply a preference for higher number of inversions, whereas the strictly positive value of δ yield a small number of inversions. $\delta = 0$ is a special case and it represents the case when the number of inversions is uniformly distributed in the interval $[0, MAXINV]$. A random number of inversions R was chosen, based on the distribution $\mathcal{Q}$. This number, R, was then randomly partitioned into an array of integers $[b_i | i = 1, 2, \ldots, M]$ such that $\sum_{i=1}^{M} b_i = R$. Following the procedure outlined in [2, pg.11-12], this array b was used as the ***inversion table*** to generate the unique data sequence that it corresponds to. Using each such sample data sequence, statistics of the form ${}_iN_j$ were updated. This procedure was repeatedly invoked to generate an ensemble of 10,000 samples. After the ensemble was obtained, the MLE of the parameter ρ was obtained by solving (11).

Table 1 shows the results for the case in which $M = 8$. Figure 1 and Figure 2 show similar results for the cases in which M equals 16 and 24 respectively.

Observe that the value of ρ increases monotonically with the increase in the expected number of inversions. When the list is almost sorted, the values of ρ are close to 0. On the other hand, as the expected number of inversions rises to values close to $M * (M-1)/4$, we find that the estimate of ρ approaches unity. This is just as is expected. The use of ρ as a presortedness metric for the ensemble is obvious.

5 Conclusions

Adaptive sorting algorithms have focussed on assigning a "presortedness" measure to specific input sequences and utilizing this information to improve the time complexity. These metrics, however, cannot be used in isolation to objectively compare the expected behaviour

[2] The reader must take care not to confuse between $\mathcal{Q}$ and $\mathcal{S}$. $\mathcal{Q}$ is the distribution on the number of inversions. $\mathcal{S}$ is the distribution for the data generation model.

δ	Expected number of Inversions	MLE of parameter ρ
0.9	0.1111	1.592e-2
0.8	0.25	3.610e-2
0.6	0.66666	1.002e-1
0.5	0.9995	1.563e-1
0.3	2.2617	3.445e-1
0.2	3.4529	4.751e-1
0.0	7.0	6.918e-1
-0.1	8.72107	7.708e-1
-0.2	10.041	8.214e-1
-0.5	12.034	9.029e-1
-0.7	12.5767	9.229e-1
-0.9	12.889	9.3619e-1

Table 1: This table shows the increase in the MLE of the parameter ρ as the Expected number of inversions in the ensemble of data sequences increases. In this case, the number of elements in each data sequence is $M = 8$. The maximum of inversions allowed is 14.

of two sorting algorithms. This requires the knowledge of the probability distribution on the permutations of the data set.

In this paper we have presented a technique to parametrize the distribution of data sequences presented to the sorting algorithm. We have proposed a presortedness metric, ρ, for an **ensemble** of data sequences, as opposed to a **single** sequence. This metric captures the divergence of data sequences in the given ensemble, from the sorted list. Various properties of the metric have been proven and a procedure for obtaining its Maximum Likelihood Estimate for a given ensemble has been described. Experimental results which support the use of the metric have been included in the paper.

References

[1] C.R. Cook and D.J. Kim, "Best Sorting Algorithms for Nearly Sorted Lists", *Comm. ACM*, Vol.23, No.11, pp.620-624, 1980.

[2] D.E. Knuth, *The Art Of Computer Programming: Vol.3 Sorting and Searching*, Addison Wesley, 1979.

[3] C. Levcopoulos and O. Petersson, "Heapsort — Adapted for Presorted Files", *Proc. of the WADS Workshop, Ottawa, Canada, August 1989*, Springer-Verlag, pp.499-509.

[4] H. Mannila "Measures of Presortedness and Optimal Sorting Algorithms", *IEEE Tranas. Computers*, Vol.C-34, No.4, pp.318-325, 1985.

[5] K. Mehlhorn, "Sorting Presorted Files", *Proc. of the Fourth GI Conf. on Theoretical Computer Science*, pp.199-212, Springer-Verlag, 1979.

[6] K. Mehlhorn, *Data Structures and Algorithms : Vol. 1: Sorting And Searching*, Springer-Verlag, Berlin/Heidelberg, F.R. Germany, 1984.

[7] B.J. Oommen and D.T.H. Ng, "Arbitrarily Distributed Random Permutation Generation", *Proceedings of 1989 ACM Comp. Sci. Conf.*, Louisville, KY, Feb 1989, pp. 27-32. Also available as Technical Report SCS-TR-138 from the School Of Computer Science, Carleton University, Ottawa, Canada, K1S 5B6.

[8] R.S. Valiveti, Ph.D. thesis, Carleton University, In preparation.

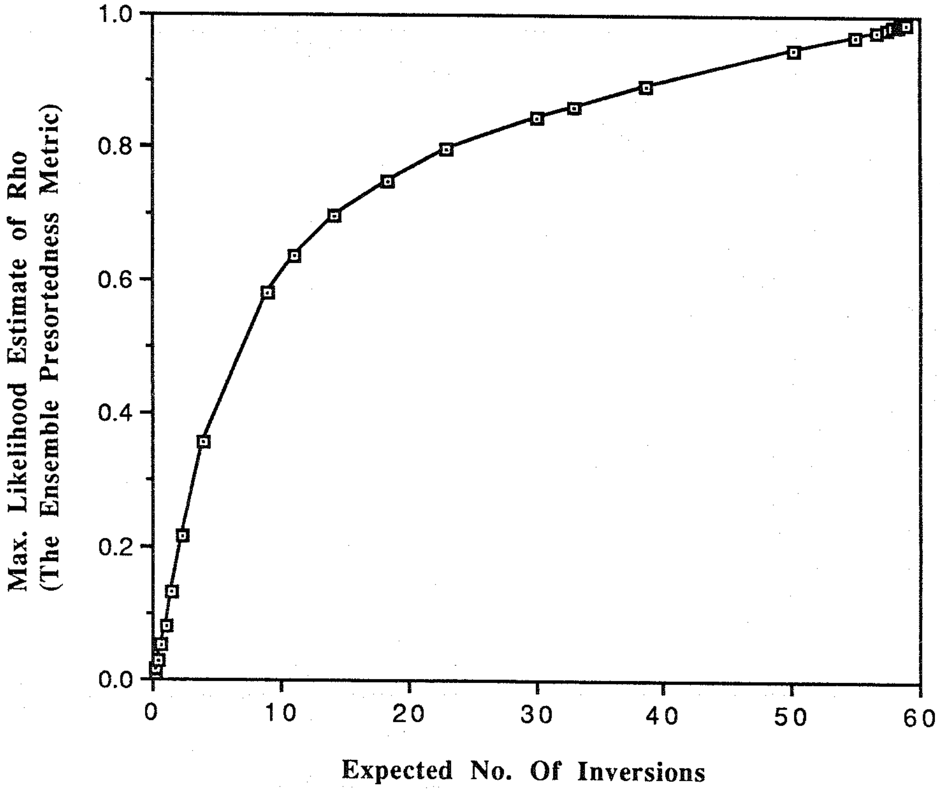

Figure 1: Figure shows the increase in the MLE of ρ as the Expected number of inversions increases. In this case $M = 16$. The maximum number of inversions allowed is 60.

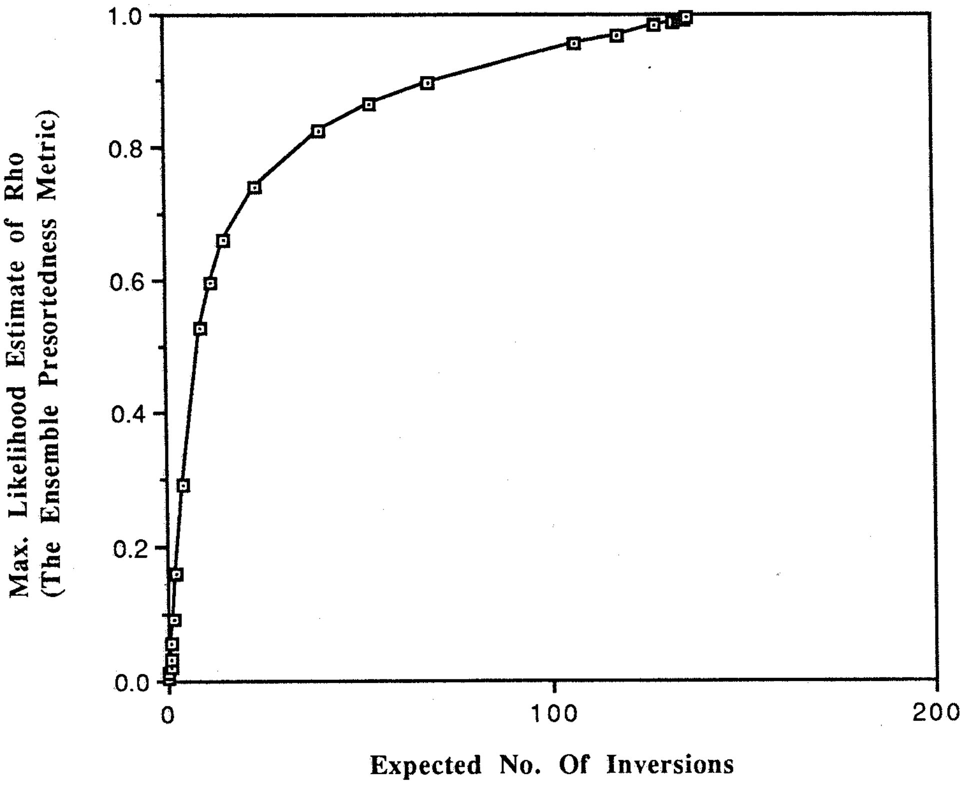

Figure 2: Figure shows the increase in the MLE of ρ as the Expected number of inversions increases. In this case $M = 24$. The maximum number of inversions allowed is 138.

School of Computer Science, Carleton University
Bibliography of Technical Reports

SCS-TR-133 **NARM: The Design of a Neural Robot Arm Controller**
Daryl H. Graf and Wilf R. LaLonde , April 1988.

SCS-TR-134 **Separating a Polyhedron by One Translation from a Set of Obstacles**
Otto Nurmi and Jörg-R. Sack, December 1987.

SCS-TR-135 **An Optimal VLSI Dictionary Machine for Hypercube Architectures**
Frank Dehne and Nicola Santoro, April 1988.

SCS-TR-136 **Optimal Visibility Algorithms for Binary Images on the Hypercube**
Frank Dehne, Quoc T. Pham and Ivan Stojmenovic, April 1988.

SCS-TR-137 **An Efficient Computational Geometry Method for Detecting Dotted Lines in Noisy Images**
F. Dehne and L. Ficocelli, May 1988.

SCS-TR-138 **On Generating Random Permutations with Arbitrary Distributions**
B. J. Oommen and D.T.H. Ng, June 1988.

SCS-TR-139 **The Theory and Application of Uni-Dimensional Random Races With Probabilistic Handicaps**
D.T.H. Ng, B.J. Oommen and E.R. Hansen, June 1988.

SCS-TR-140 **Computing the Configuration Space of a Robot on a Mesh-of-Processors**
F. Dehne, A.-L. Hassenklover and J.-R. Sack, June 1988.

SCS-TR-141 **Graphically Defining Simulation Models of Concurrent Systems**
H. Glenn Brauen and John Neilson, September 1988

SCS-TR-142 **An Algorithm for Distributed Mutual Exclusion on Arbitrary Networks**
H. Glenn Brauen and John E. Neilson, September 1988

SCS-TR-143 to 146 are unavailable.

SCS-TR-147 **On Transparently Modifying Users' Query Distributions**
B.J. Oommen and D.T.H. Ng, November 1988

SCS-TR-148 **An O(N Log N) Algorithm for Computing a Link Center in a Simple Polygon**
H.N. Djidjev, A. Lingas and J.-R. Sack, July 1988
Available in STACS 89, 6th Annual Symposium on Theoretical Aspects of Computer Science, Paderborn, FRG, February 16-18, 1989, Lecture Notes in Computer Science, Springer-Verlag No. 349

SCS-TR-149 **Smallscript: A User Programmable Framework Based on Smalltalk and Postscript**
Kevin Haaland and Dave Thomas, November 1988

SCS-TR-150 **A General Design Methodology for Dictionary Machines**
Frank Dehne and Nicola Santoro, February 1989

SCS-TR-151 **On Doubly Linked List ReOrganizing Heuristics**
D.T.H. Ng and B. John Oommen, February 1989

SCS-TR-152 **Implementing Data Structures on a Hypercube Multiprocessor, and Applications in Parallel Computational Geometry**
Frank Dehne and Andrew Rau-Chaplin, March 1989

SCS-TR-153 **The Use of Chi-Squared Statistics in Determining Dependence Trees**
R.S. Valiveti and B.J. Oommen, March 1989

SCS-TR-154 **Ideal List Organization for Stationary Environments**
B. John Oommen and David T.H. Ng, March 1989

SCS-TR-155 **Hot-Spot Contention in Binary Hypercube Networks**
Sivarama P. Dandamudi and Derek L. Eager, April 89

SCS-TR-156 **Some Issues in Hierarchical Interconnection Network Design**
Sivarama P. Dandamudi and Derek L. Eager, April 1989

SCS-TR-157 **Discretized Pursuit Linear Reward-Inaction Automata**
B.J. Oommen and Joseph K. Lanctot, April 1989

SCS-TR-158 (revised) **Parallel Fractional Cascading on a Hypercube Multiprocessor**
Frank Dehne, Afonso Ferreira and Andrew Rau-Chaplin, May 1989 (Revised April 1990)

SCS-TR-159 **Epsilon-Optimal Stubborn Learning Mechanisms**
J.P.R. Christensen and B.J. Oommen, June 1989

SCS-TR-160 **Disassembling Two-Dimensional Composite Parts Via Translations**
Doron Nussbaum and Jörg-R. Sack, June 1989

SCS-TR-161 (revised) **Recognizing Sources of Random Strings**
R.S. Valiveti and B.J. Oommen, January 1990
Revised version of SCS-TR-161 "On the Data Analysis of Random Permutations and its Application to Source Recognition", published June 1989

SCS-TR-162 **An Adaptive Learning Solution to the Keyboard Optimization Problem**
B.J. Oommen, R.S. Valiveti and J. Zgierski, October 1989

SCS-TR-163 **Finding a Central Link Segment of a Simple Polygon in O(N Log N) Time**
L.G. Alexandrov, H.N. Djidjev, J.-R. Sack, October 1989

SCS-TR-164 **A Survey of Algorithms for Handling Permutation Groups**
M.D. Atkinson, January 1990

SCS-TR-165 **Key Exchange Using Chebychev Polynomials**
M.D. Atkinson and Vincenzo Acciaro, January 1990

SCS-TR-166 **Efficient Concurrency Control Protocols for B-tree Indexes**
Ekow J. Otoo, January 1990

SCS-TR-167 **A Hierarchical Stochastic Automaton Solution to the Object Partitioning Problem**
B.J. Oommen, January 1990

SCS-TR-168 **Adaptive List Organizing for Non-stationary Query Distributions. Part I: The Move-to-Front Rule**
R.S. Valiveti and B.J. Oommen, January 1990

SCS-TR-169 **Trade-Offs in Non-Reversing Diameter**
Hans L. Bodlaender, Gerard Tel and Nicola Santoro, February 1990

SCS-TR-170 **A Massively Parallel Knowledge-Base Server using a Hypercube Multiprocessor**
Frank Dehne, Afonso Ferreira and Andrew Rau-Chaplin, April 1990

SCS-TR-171 **Parallel Processing of Quad Trees on the Hypercube (and PRAM)**
Frank Dehne, Afonso Ferreira and Andrew Rau-Chaplin, April 1990

SCS-TR-172 **A Note on the Load Balancing Problem for Coarse Grained Hypercube Dictionary Machines**
Frank Dehne and Michel Gastaldo, May 1990

SCS-TR-173 **Self-Organizing Doubly-Linked Lists**
R.S. Valiveti and B.J. Oommen, May 1990

SCS-TR-174 **A Presortedness Metric for Ensembles of Data Sequences**
R.S. Valiveti and B.J. Oommen, May 1990