Packed Permutations and Inversions: Properties and Applications.

Fabio Guerinoni

Mathematics and Computer Science Department
Bemidji State University

Abstract

In this paper, we present a bit-based method to store permutations. From the standard representation, we introduce *order matrices* which relates the well-known inversion tables and provides the packed representation. Some basic algorithms are presented to manipulate these objects. We briefly mention some applications that may advantageously use the global information that they disclose.

Keywords: permutation, cycles, inversions, shuffles, parallel procesing.

1. Introduction

Consider a set of values, ranking certain data which is distributed among a number of computers (processors). The data and their ranking was produced over some period of time and assigned to the computers sequentially. If some processor wants to know if it holds data that is larger in value than anything that came previously, it seems that it will need to communicate with the other processors. It seems that the problem is *inherently sequential*.
This is not the case if the permutation is represented as an *inversion table*, as we will see later. Inversion tables were discovered, apparently relatively late, by Marshall Hall Jr. in 1956 (Knuth, 1998), although the concept of *inversion of a permutation*, has been in the literature for longer in connection with the theory of determinants.
We are all familiar with a permutation π, belonging to the symmetric group of order n, $\mathcal{S}_n$, a one-to-one mapping of the set $\{1, 2, \ldots, n-1, n\}$ into itself, often denoted

$$\begin{pmatrix} 1\ 2\ \ldots n-1\ n \\ p_1\ p_2\ \ldots p_{n-1}\ p_n \end{pmatrix}$$

where $\pi(i) = p_i$. Once we have this convention, we may simply write π as $\mathbf{p}(\pi) = (p_1\ p_2\ \ldots p_{n-1}\ p_n)$: the argument of the mapping is given implicitly by position. This *one-line* notation is often used (Knuth 1998), but there is no reason why we can't describe π in any other way, say by writing the complement: $\mathbf{p^c}(\pi) = (n+1-\pi(1), \ldots, n+1-\pi(n-1), n+1-\pi(n))$. Although it is important to distiguish the permutation, from its *representation*, in this article, at times, we blur this difference by writing $\pi(i) = p_i = \pi_i$.
One advantage of alternative representations is that they may convey non-local information and thus may be useful in many applications and algorithms. In practical, as well as in theoretical computer science, the issue of representation is of paramount importance. However, this non-locality makes algorithms for complex problems simple, and algorithms for simple problem complex.
One such representation are the inversion tables. In Section 2, we review their basic properties, and we introduce notation for further sections. Section 3 is the main section and it introduces the concept of *packed permutation*, and after giving some framework, we derive some simple properties, including its basic relations with the inversions tables.
Section 4 describes two algorithms to construct packed representations and to operate with them. The algorithms have a rich structure that can be subject to analysis, but this is not explored here. Finally, Section 5 outlines some ideas for applications to certain types of permutations and to distributed processing.

2. Inversion Tables

Consider a set of values, ranking certain data which is distributed among a number of computers (processors). The data and their ranking was produced over some period of time and assigned to the computers sequentially.

As we stated before, inversions tables have been known for for long, We will introduce them here. First, we define some notation.
It is common to study a permutation π by its effect on any order set of n elements. We allow a permutation to take as 'argument' a n-vector (totally ordered set) and produce another n-vector arranged according to the permutation. Thus if $s = (s_1\ s_2\ \ldots s_{n-1}\ s_n)$, then

$$s^{\pi} = (s_{\pi^-(1)}, s_{\pi^-(2)} \ldots s_{\pi^-(n-1)}, s_{\pi^-(n)})$$

and we say that π has *acted* on s. Alternatively, the effect of a permutation on an ordered set can be interpreted as *vector-matrix product*. In what follows, we use the superscript t, as in U^t, to denote vector or matrix transpositon.

Definition 1.1. The matrix $\Pi = [\mathbf{e}_{\pi(1)}, \mathbf{e}_{\pi(2)} \ldots \mathbf{e}_{\pi(n)}]$ with

$$\mathbf{e}_i^t = (0, \ldots 1, \ldots 0, 0)$$

where the single 1 occurs at position i. is called the permutation matrix of π. Clearly if an n-vector s in column form is multiplied on the right by Π, the resulting row vector is the same as s^{π}.
The inverse of π will be denoted more simply by π^-, rather than using the cumbersome -1 exponent; the one line representation for π^- will be denoted $\mathbf{p}^-$.
Using # to denote the cardinality of a set, the *number of inversions* in a permutation is defined

$$\mathrm{inv}(\pi) = \#\big\{\ (i,j)\ /\ i < j, p_j < p_i\big\}.$$

In other words, using the standard representation $\mathbf{p}$, we are counting the number of pairs when p_i occurs left of p_j and p_i is also greater than p_j. We are using the standard representation to denote the permutation π; we follow the same practice throughout the paper for notational convenience.
The two inversion representations for π, $\mathbf{q}(\pi)$, $\overline{\mathbf{q}}(\pi)$ arise from counting inversions in orderly ways:

$$q_i = \#\Big\{ j\ /\ j < i,\ p_i < p_j \Big\} \tag{1}$$

$$\overline{q}_i = \#\Big\{ j\ /\ j < p_i^-,\ i < p_j \Big\} \tag{2}$$

Either one defines an inversion table, but Hall in his original paper, used (2).
Using simple constructive arguments, it can be shown that $\overline{\mathbf{q}}$ define π, and also $\mathbf{q}$ as we will see in short.
For example, we take the permutation τ whose standard respresentation is

$$\mathbf{p}(\tau) = (1\ 7\ 3\ 2\ 8\ 4\ 6\ 9\ 5)$$

that is $\tau(1) = 1, \tau(2) = 7, \tau(3) = 3 \ldots$. Then

$$\mathbf{q}(\tau) = (\ 0\ 0\ 1\ 2\ 0\ 2\ 2\ 0\ 4) \qquad \overline{\mathbf{q}}(\tau) = (0\ 2\ \ 1\ 2\ 4\ 2\ 0\ 0\ 0)$$

Proposition 2.1. If $\mathbf{q}$ and $\overline{\mathbf{q}}$ are the two inversion tables corresponding to π, (1), (2). Then,

$$\mathbf{q}^{\pi} = \overline{\mathbf{q}}$$

Proof: This follows directly from the definition, but it is illustrative to revert to the two-line notation. If $q_i = r$ there are indices $j_1 < j_2 < \ldots < j_r < i$ with $p_i < p_{j_k},\ \forall k\ 1 \le k \le r$. The two line notation for π is

$$\begin{pmatrix} \ldots\ j_1\ \ldots\ j_2\ \ldots j_r \ldots\ i \ldots n \\ \ldots\ p_{j_1} \ldots\ p_{j_2}\ \ \ldots p_{j_r} \ldots p_i \ldots p_n \end{pmatrix}$$

from where we can clearly see that $\overline{q}_{p_i} = r$. Thus $\overline{q}_{p_i} = q_i$. And since $p_i = \pi(i)$, $\overline{q}_i = q_{\pi^-(i)}$.

Proposition 2.2. For each of the inversion representations, $\mathbf{q}, \overline{\mathbf{q}}$, the following bounds are observed:

$$0 \leq q_i \leq i-1 \qquad 0 \leq q_i \leq n - p_i \tag{3}$$

$$0 \leq \overline{q}_i \leq n-i \qquad 0 \leq \overline{q}_i \leq p_i^- - 1 \tag{4}$$

Proof: The first line is immediate by the definition. The second line is obtained by substituting $\pi^-(i)$ for i and using Proposition (1.1). ∎

For each inversion table, the first bounds are absolute, while the second type of bound depends on the permutation π represented. In general, the first bound gives $\mathbf{q}$, an "increasing" appearance, and $\overline{\mathbf{q}}$ a "decreasing" one. So sometimes we will refer to the two inversions as the *up vector* and the *down vector* of π.

By taking any of the two range inequalities in the first column in 3 and 4, it is not hard to see that there are exactly $n!$ inversion tables that will satisfy them. Also, the $\overline{q}$ may be used to naturally map the symmetric group to $\{0, 1, \ldots, n!-1\}$ using a 'factorial base.' This fact is often used for enumeration and the generation of random permutations.

Definition 2.1. For $\pi \in \mathcal{S}_n$. $\pi(i)$ is a *left to right maximum*, (LR-max) if $\pi(i)$ is the largest value among the $\pi(k), 1 \leq k \leq i$. The LR-max *occurs* at i. Similar definitions can be given for LR-min, for the right to left versions, RL-max, RL-min.

Notice that we have a zero value in $\mathbf{q}$ wherever a LR-max occurs in π. Wherever $\overline{\mathbf{q}}$ is zero, tells the *value* of a LR-max. Other similar properties can be derived; some are summarized here.

Proposition 2.3.

(a) π has a LR-max (min) at $i \Leftrightarrow q_i = 0$ ($q_i = i-1$).
(b) i is a LR-max (RL-max) $\Leftrightarrow \overline{q}_i = 0$ ($\overline{q}_i = n-i$)

Proof: For part (a), it is almost by definition: if $q_i = 0$ there are no larger p_j left of p_i, so it must be a record maximum. $q_i = i-1$, the p_j are all larger, so it must be a record minimum. Part (b): by Proposition 1 if $\overline{q}_i = 0$ if and only if $q_{\pi^-(i)} = 0$, this means that for all indices $j < \pi^-(i)$ we have $p_{\pi^-(i)} = i > p_j$, that is, i is a LR-max. For the second part, a less formal argument will be used. If $\overline{q}_i = n-i$, all the values larger than i occur left of i in $\mathbf{p}$. This can only happen, if i is larger than anything to the right. All the arguments are reversible. ∎

The following show the relation of the inversions tables of π with its inverse.

Proposition 2.4.

(a) p_k is an LR-maximun for $\pi \Leftrightarrow \pi^-$ has a RL-minimum *at* k .
(b) π^- has a RL-min (max) at $i \Leftrightarrow \overline{q}_i = 0$ ($\overline{q}_i = n-i$).

Proof: Similar to for Proposition (2.3). ∎

3. A packed representation of a permutation

Consider a set of values, ranking certain data which is distributed among a number of computers (processors). The data and their ranking was produced over some period of time and assigned to the computers sequentially.

To show the relations between the two inversions we introduce the concept: *the order matrix*. It is defined to be

$$P_{ij}(\pi) = [\pi(i) < \pi(j)]$$

where we use $[q(i,j)]$ is characteristic function of the predicate q. Note that we have made the order matrix independent of the representation but the standard definition could have been used in the representation.

In what follows, the standard representation and the inversion tables will be assumed to be a column n-vector. We will use the $\mathbf{1}$ to the n-vector column composed of 1. It is easy to see the $\mathbf{1}\ \mathbf{1}^t$ is the matrix composed just of 1. We will call I', the permutation matrix representing reversion, that is with ones on the secondary diagonal and zero elsewhere. $\mathbf{n}$ will the column n-vector consisting of $1, 2, \ldots n$. In other words, $\mathbf{n} = \mathbf{p}(id)$

The simplest property which can be directly deduced from the definition is

$$P + P^t = \mathbf{1}\mathbf{1}^t - I,$$

tells us that the order matrix represents a *tournament*, with the additional property of being transitive. The no-ties outcome of the tournament defines the permutation.
If we fix a row i and sum over j, we account for the elements larger than p_i; similarly, fixing a column j and summing over i, we must get exactly $p_j - 1$. Thus,

$$\begin{aligned} P\mathbf{1} &= n\mathbf{1} - \mathbf{p} \\ \mathbf{1}^T P &= \mathbf{p}^T - \mathbf{1}^T \end{aligned}$$

Thus the columm sum of P "complement" minus 1, as defined in the introduction, and the row sum gives us the original representation $\mathbf{p}$ minus 1.
We decompose P as a sum of a stricly lower triangular and a strictly upper triangular matrix.

$$P(\pi) = L(\pi) + U(\pi)$$

The symmetric matrices, $L+L^T$ and $U+U^T$ are very well known in the literature, as being the adjacency matrices of *permutation graphs*. (Golumbic,1980) . Permutation graphs are of interest in theoretical computer science, because often they admit tractable versions of algorithms which are normally intractable. The classic characterization of such graphs is given in (Even et al,1972): they can be labeled so that the adjacency matrices of *both* the graph (L) and its complement (U) are transitive.

Definition 3.1. The *packed* representation of the permutation π is given by the $n(n+1)/2$ bits represented by $L(\pi)$. This include n redundant 0-bits from the diagonal which are kept for convenience of description.
For example the permutation $\mathbf{p} = (5, 3, 1, 6, 2, 4)$ has the following packed representation:

$$\begin{pmatrix} 0 & & & & & \\ 1 & 0 & & & & \\ 1 & 1 & 0 & & & \\ 0 & 0 & 0 & 0 & & \\ 1 & 1 & 0 & 1 & 0 & \\ 1 & 0 & 0 & 1 & 0 & 0 \end{pmatrix}$$

By construction, a permutation can only have one packed representation.
The following result is important and shows our proposed representation as the link showing the duality between the two types of inversions.

Proposition 3.1. Given $\pi \in \mathcal{S}_n$

$$\begin{aligned} \mathbf{q}(\pi) &= L(\pi)\ \mathbf{1} \\ \overline{\mathbf{q}}(\pi^-) &= L^t(\pi)\ \mathbf{1} \end{aligned}$$

Proof: The relation between L and $\mathbf{q}$ is straightforward. We have

$$q_i = \sum_{j<i} [p_i < p_j]$$

For part (b), we transpose both sides and look at the j element

$$(\mathbf{1}^t L)_j = \sum_{j<i} [p_i < p_j]$$

if there is any non-zero terms, we have

$$\begin{pmatrix} \ldots\ldots j \ldots & i \ldots n \\ \ldots\ldots p_j & \ldots p_i \ldots p_n \end{pmatrix}$$

where $p_i < p_j$. Upon inverting, we get

$$\begin{pmatrix} \ldots\ldots p_i \ \ldots p_j \ldots p_n \\ \ldots\ldots i \ldots \ j \ldots n. \end{pmatrix}$$

Thus, there will be a corresponding contribution to $\overline{\mathbf{q}}(\pi^-)$. ∎

Definition 3.2. We call $\mathcal{U}_n$, the *up-space* of order n, the set of vectors $\mathbf{u} = (u_i, \ldots u_n)$ satisfying $0 \leq u_i \leq i-1$. An element $\mathbf{d}$ of $\mathcal{D}_n$, the *down-space*, is defined in a similar way but satisfying $0 \leq d_i \leq n-i$.
Proposition (2.2) shows that for any π, $\mathbf{q}(\pi) \in \mathcal{U}_n$ and $\overline{\mathbf{q}}(\pi) \in \mathcal{D}_n$. Conversely, to each element of $\mathcal{U}_n$ ($\mathcal{D}_n$) we can associate a unique permutation in $\mathcal{S}_n$. For that reason, we will often use $\mathbf{q}^{\mathcal{U}}$ and $\mathbf{q}^{\mathcal{D}}$ to denote, respectively, $\mathbf{q}$ and $\overline{\mathbf{q}}$
Proposition (3.1) also shows a natural relation, via the packed representation, between elements in the up-space and the down-space, that merit its own definition.

Definition 3.3. $u \in \mathcal{U}_n$ and $d \in \mathcal{D}_n$ are said to be dual of each other if $\exists \pi \in \mathcal{S}_n$with $\mathbf{u} = \mathbf{q}^{\mathcal{U}}(\pi)$ and $\mathbf{d} = \mathbf{q}^{\mathcal{D}}(\pi^-)$. In each case π is the *underlying permutation*. We write, $\mathbf{u}^* = \mathbf{d}$ and $\mathbf{d}^* = \mathbf{u}$.
Proposition (2.2) gives the range of $\mathbf{q}$ and $\overline{\mathbf{q}}$. Then we can naturally define complements (componentwise to the their maximum of their range), similar to the definition of $\mathbf{p}^c$ in Section 1. For example,

$$\begin{aligned} \mathbf{q}(\pi) + \mathbf{q}^c(\pi) &= L\mathbf{1} + U^t\mathbf{1} && (5) \\ &= \mathbf{n} - \mathbf{1} && (6) \end{aligned}$$

Definition 3.4. Given $\mathbf{q}$ in $\mathcal{U}_n$ or $\mathcal{D}_n$ we define the following operators whenever they are valid:

$$^{++}\mathbf{q}(i) = [q_1, q_2, \ldots, q_i + 1 \ldots, q_n] \qquad ^{--}\mathbf{q}(j) = [q_1, q_2, \ldots, q_j - 1 \ldots, q_n]$$

and we call them elementary increment(decrements). The operators are not defined if the resulting value is not in the respective space.

Definition 3.5. We define

$$\mathrm{glb}_i(\mathbf{p}) = \max_{1 \leq j \leq i} (p_j; p_j < p_i) \qquad \mathrm{lub}_i(\mathbf{p}) = \min_{1 \leq k \leq i} (p_k; p_k > p_i)$$

Proposition 3.2. Let π be such that $\mathbf{q}^{\mathcal{U}}(\pi) = \mathbf{q}$. The permutation τ corresponding to $^{++}\mathbf{q}(i)$ is obtained by exchanging element π_i with the element $\mathrm{glb}(\pi_i)$. We will call this a *minimum inversion increase* exchange at i.
Proof: : to see this , let j be the index corresponding to the $\mathrm{glb}(\pi_i)$, that is $j = \arg \mathrm{glb}(\pi_i)$. After the exchange, $\tau_i = \pi_j, \tau_j = \pi_i$, all other entries of τ remaining the same. Calling $\mathbf{q}' = \mathbf{q}^{\mathcal{U}}(\tau)$. Clearly, $q'_i = q_i + 1$. Also the only indices that could have been possible affected are k, with $j \leq k < i$. The q_j cannot change. For index j, any element larger than π_j will still be larger than π_i. For $k \neq j$, q_k cannot increase, as a larger element is brought to the left. If assume that q_k increases, that would mean $\pi_i > \pi_k$ but $\pi_j < \pi_k$, contradicting the choice of j. ∎

In a similar way, the following can be proved.

Proposition 3.3. : The permutation corresponding to $^{--}\mathbf{q}(j)$ is obtained by exchanging element a position with element $\mathrm{lub}(\pi_i)$, where the choice is made among the elements of π with index less than i.

If $\mathbf{q} = \mathbf{q}(\pi)$. Proposition (3.2) shows how to compute π' such that it corresponds to $^{++}\mathbf{q}(i)$:

$$\mathbf{p}' = (p_i \ \mathrm{glb}_i(\mathbf{p}))\ \mathbf{p}$$

where the parenthesis represents cycle notation for transpositions. The non-local nature of the elementary increment is quantifyied. The next proposition is a similar result for the down vectory.

Proposition 3.4. Consider a down-vector $\mathbf{d} = (i \ d_2 \ \ldots d_{n-1} \ d_n)$ with dual $\mathbf{d}^* = \mathbf{c} = (c_1 \ c_2 \ \ldots c_{n-1} \ c_n)$ Assume $0 \leq i < n-1$. Call,

$$\mathbf{d}_+ =^{++} \mathbf{d}(1) = (i+1\ d_2\ \ldots d_{n-1}\ d_n)$$

Then,

$$\mathbf{d}^*_+ = \{c_1\ c_2\ \ldots c_{\pi^-(i+2)} + 1\ \ldots\ c_n\}$$

Proof: Notice that since $i < n-1$, $i+1$ and $i+2$ are defined in the context where they are used. Call τ the underlying permutation for $\mathbf{d}_+$. We will show that $\mathbf{q}^{\mathcal{U}}(\tau) = \mathbf{d}^*_+$. Let π be the underlying permutation for $\mathbf{d}$. π^- has 1 at position $i+1$. Thus, τ^- is obtained from π^- by transposing 1 with the element immediately to its right:

$$\tau^- = \begin{pmatrix} 1 & 2 & \ldots & i & i+1 & i+2 & \ldots & n \\ \pi_1^- & \ldots & \ldots & \pi_i^- & \pi_{i+2}^- & 1 & \ldots & \pi_n^- \end{pmatrix}$$

Inverting we get,

$$\tau = \begin{pmatrix} 1 & \ldots & \pi_{i+2}^- & \ldots n \\ i+2 & \ldots & i+1 & \ldots \pi_n \end{pmatrix}.$$

If we compare with π

$$\pi = \begin{pmatrix} 1 & \ldots & \pi_{i+2}^- & \ldots n \\ i+1 & \ldots & i+2 & \ldots \pi_n \end{pmatrix},$$

we see that τ is obtained from π by a minimum inversion increase exchange at π_{i+2}^-. By Proposition (3.3), $\mathbf{d}^*_+$ must have the form stated.

4. Some Computational Aspects

Consider a set of values, ranking certain data which is distributed among a number of computers (processors). The data and their ranking was produced over some period of time and assigned to the computers sequentially.
In the previous sections, we saw explicit form for some elementary operations on permutation and its effects on that inversion tables. Corresponding effects on the packed permutations are more difficult to describe symbolically, and we use instead algorithms.
The proposed representation of a permutation π uses the lower triangular part of $P(\pi)$ which contains all the information in P in view of the antisymmetry (). We must have means to quickly operate on these so as to verify the homomorphism between $\mathcal{S}_n$and $\mathcal{U}_n$ (or $\mathcal{D}_n$). We might try a matrix product:

$$P(\pi\tau) = P(\pi)\ P(\tau) \tag{7}$$

where the right hand product is defined in the natural way: calling $C = PQ$

$$C_{i,j} = P_{i,1}Q_{1,j} + P_{i,2}Q_{2,j} + \ldots + P_{i,n}Q_{n,j} \tag{8}$$

We need at least a ring structure in $\{0,1\}$, to play the roles of the additive group and associative operators. Unfortunately there is not much choice. The only candidates for the additive group operators are given in the following table, with their respective identity element. There is only such four rings, all involving the logic operations:

$\oplus$	$\equiv$	$\vee$	$\wedge$
0	1	0	1

(we use $\oplus$ for exclusive-or). Unfortunately, as one can easily verify for the transposition (2 1) in $\mathcal{S}_2$

Proposition 4.1. No product of the form (8) exists that satifies (7), regardless of the choice of operators.

On the other hand, the order matrices, and thus the packed permutations, behave well under *conjugation*. Conjugation in a permutation can be understood as a relabeling; the basic structure is preserved. Similarly, matrix conjugation with a permutation matrix also reflects a relabeling. This observation results in

Proposition 4.2. For $\tau \in \mathcal{S}_n$, call T the permutation matrix corresponding to τ. Then,

$$P(\tau^{-}\pi\tau) = T^t P(\pi) T$$

The rest of the section will illustrate two types of algorithms, one 'external', which are derived from constructions in $\mathcal{U}_j, j = 1, 2, \ldots n$ (or $\mathcal{D}_j$); and the other 'internal,' with fixed up or down space. The latter are more elaborated, and involves some form of inner product.
The following concepts are useful, in the construction and description of the external algorithms.

Definition 4.1. Let $\pi \in \mathcal{S}_n$with down vector $\mathbf{d} \in \mathcal{D}_n$, $\mathbf{d} = (d_1\ d_2\ \ldots d_{n-1}\ d_n)$ and up vector $\mathbf{u} \in \mathcal{U}_n$, $\mathbf{u} = (u_1\ u_2\ \ldots u_{n-1}\ u_n)$. We associate a sequence $\delta_i \in \mathcal{S}_{n-i+1},\ 1 \leq i \leq n$ to π called the *descending sequence*

$$\mathbf{q}^{\mathcal{D}}(\delta_i) = (d_{n-i+1}, d_{n-i+2} \ldots, d_2, d_1) \in \mathcal{D}_{n-i+1}$$

and another sequence $\upsilon_j \in \mathcal{S}_j$ called the *ascending sequence*

$$\mathbf{q}^{\mathcal{U}}(\upsilon_j) = (u_1, u_2 \ldots, u_{j-1}, u_j) \in \mathcal{U}_j$$

Note that $\upsilon_n = \delta_1 = \pi$.
Algorithm 1 computes υ_i and $L(\upsilon_i)$ from $\mathbf{u} \in \mathcal{U}_n$. We denote the entries of the packed representation $L(\pi)$ as $l_{k,l}$ with the tacit assumption that $k > l$. We also use C-like assignments and relational operators. The blocks are implied by the indentation.

ALGORITHM 1:
Input $\mathbf{u} \in \mathcal{U}_n$, $i \leq n$
Output : $\upsilon_i = (c_1, c_2 \ldots c_i)\ L(\upsilon_i)$
  for $(1 \leq r \leq i)$
    $c_r = r - u_r\ ;\quad l_{i,r} = 0$
  for $(1 \leq j < i)$
    if $(c_j \geq c_r)$ **then**
      $++c_j\ ;\quad l_{i,j} = 1$

  return ;

Proposition 4.3. Algorithm 1 computes $\mathbf{p}(\pi)$and $L(\pi)$ on input $\mathbf{q}^{\mathcal{U}}(\pi)$ and n.

Proof: (Sketch) The key observation is that the last entry in $\mathbf{q}(\pi)$, q_n is the complement of p_n, $n - p_n$. Now, this number also counts the values among $c_1 \ldots c_{n-1}$ greater than it . By the induction process, these values represent $\mathbf{p}_{n-1}$ and thus have range $1, 2, \ldots n-1$. The last iteration increases by one the required values, of which there are exactly q_n of them. The row n of L is marked accordingly. ∎

A similar algorithm can be defined to form the down sequence for $\mathbf{q}^{\mathcal{D}}$ and the simultaneous computation of L. This algorithm is similar to the one outlined in (Knuth, 1998) to recover permutations from inversions, with the advantage that ours does not required a linked list for insertions. As in Algorithm 1, the permutation corresponding to $\mathbf{q}^{\mathcal{D}}$ is recovered in standard form.
Now we consider the problem analogous to Proposition (3.2), but which modifies the packed representation. That is, given $\mathbf{q}^{\mathcal{U}}(\pi') =^{++} \mathbf{q}(\pi)i$ we want $L(\pi')$, granted that we know $L(\pi)$. For given i, n and L, Algorithm 2 computes the desired packed representation.
For convenience in describing the algorithm, we introduce the negative implication product for boolean vectors V, W

$$V \not\rightarrow W = \sum V_i \not\rightarrow W_i \tag{9}$$

$$= \sum \neg W_i \wedge V_i \tag{10}$$

The sum is taken in the normal arithmetic sense. Thus, there is contribution if $v_i = 1$ and $w_i = 0$.
Assuming i is the input, the vectors C_j of length $n - i$ are used to simplify the description of the algorithm and correspond to the partial column-vectors in L:

$$C_j = (l_{i+1,j}, l_{i+2,j}, \ldots l_{n,j}).$$

Algorithm 2:

Input $L(\pi)$, i
Output: Packed permutation corresponding to ${}^{++}\mathbf{q}(\pi)(i)$

$min = n - i\,;\quad ix = 0\,;$
for $(1 \le r \le i)$
  if $(l_{i,r} == 0)$ **then**
    $m = C_i \not\to C_r$
    if $(m < min)$ **then**
      $min = m\,;\quad ix = r\,;$
  if $(ix == 0)$ **then**
    return ;
$l_{i,ix} = 1$

for $(i + 1 \le k \le n)$
  if $(l_{ix,k} == 0 \ \wedge \ l_{ix,i} == 1)$ **then**
    $l_{ix,i} = 0\,;\quad l_{ix,k} = 1\,;$
return ;

Due to the natural bounds on the up vector ${}^{++}\mathbf{q}(i)$ might not be defined. This causes the algorithm to return quietly without any changes to L.

5. Some Applications

Consider a set of values, ranking certain data which is distributed among a number of computers (processors). The data and their ranking was produced over some period of time and assigned to the computers sequentially.
Here we mention two potential application of some of the concepts in this paper. Certainly there are many more. We outline only the main ideas, the practicality of them and the performance in actual applications remain to be explored further.

Shuffle permutations. We need some preliminary definitions.

Definition 4.2. A *rising sequence* in the permutation π is a set of consecutive numbers $(q \ \ q{+}1 \ \ldots \ q{+}l{-}2 \ \ q{+}l{-}1)$ such that

$$\pi^-(q) < \pi^-(q+1) < \ldots < \pi^-(q+l-2) < \pi^-(q+l-1)$$

and furthermore satisfying $\pi^-(q-1) > \pi^-(q)$ and $\pi^-(q+l-1) > \pi^-(q+l)$.
The last conditions assure that a rising sequence is *maximal*, that is they cannot be strictly contained in another rising sequence.
A particular type of permutation is that one that contains exactly two rising sequences. Such permutations are called *shuffles* by analogy of cutting and mixing a deck of cards. Shuffles are very well study, because of their application in cryptography, and the design of sorting networks and computer memories.
For simplicity we assume that to represent a number k, we need $\lg k$ bits Using $p(\pi)$,

$$R(\pi) = \sum_{i=1}^{n-1} \lg i \tag{11}$$

bits of information for each of them.

Proposition 5.1. For a shuffle $\sigma \in \mathcal{S}_n$, we have

$$\mathbf{q}^{\mathcal{D}}(\pi) = (w_1\ w_2\ \ldots w_k\ 0\ 0\ \ldots 0\ 0)$$

with $w_1 <= w_2 \ldots <= w_k = k-1$.
Thus shuffles can be stored in smaller arrays. The advantages of representing it as an inversion is immediately apparent. Thus, each w_i will require no more than $\lg(k-1)$ bits, under our assumption. And we can easily bound the number of bits required:

$$R(\sigma) = \sum_{i=1}^{k-1} \lg w_i \leq (k-1)\lg(k-1)$$

Let R_{avg} the number of bits required for the *average* shuffle. Under one common model (Bayer and Diaconis, 1992), in which the 'cut' (represented by k) is made uniformly random, the average shuffle will have $k = n/2$, thus

$$R_{avg} \leq \frac{n}{2}\ \ln n/2$$

For large n, an asymptotic expression can be used for (11) :

$$\ln 2R(\pi) = n\ln n - n + \sqrt{2\pi} - \frac{\ln 2}{2} + O(1/n)$$

Then,

$$\begin{aligned} R(\pi)/R_{avg} &\geq R(\pi)/(\frac{n}{2}\ \ln n/2) \\ &\geq \frac{2}{\ln 2} \approx 2.88541 \end{aligned}$$

Thus, using the $\mathbf{q}^{\mathcal{D}}$ representation for shuffles is takes three times less space, on average. While this is not surprising because shuffles convey less information, (the information-theoretical ratio is $O(\ln n)$), it suggests using inversion tables or packed representations.

Distributed Processing. This, I hope, will answer the question posed in the Introduction. Considering an application, like for example in very large data bases, where the (real-time) data "wagers" is arriving sequentially to processors (or resources) $0 \ldots M-1$. Each resource may store up to n items and keeps a subarray of global ranking indices p_i. This array is distributed (due to hypothetical space limitations), that is, processor r stores $p_{nr+1} \ldots p_{(r+1)n}$.
A query is received by all processors simultaneously. Any query involving a given global rank, (for example: Which wager was the maximum? minimun?, fifth largest?) is resolved by locally searching the n array of data, and each processor/resource may work independently. But now consider other global questions like: Which wagers surpassed all previous ones? Or the previous 100? The above discussion on the properties of inversions/packed representations, in particular Propositions 2.3 and 2.4, should make clear that that information may be recovered from data local to the processor alone.

6. Authorization and Disclaimer

Authors authorize LACCEI to publish the papers in the conference proceedings. Neither LACCEI nor the editors are responsible either for the content or the implications of what is expressed in the paper.

7. References

Bayer,D. and Diaconis,P.,1992 "Trailing the dovetail shuffle to its lair". *Annals of Appl. Prob."*, 2 (2):294–313
Even, S., Lempel,A. and Pnueli,A., 1972 "Permutation graphs and Transitive graphs."*J. ACM*, 19:400–410,
Golumbic, M., 1980 "Algorithmic Graph Theory and Perfect Graphs." Academic Press
Knuth, D., 1998. Sorting and Searching,"The Art of Computer Programming."Vol 3, AddissonWesley, 3rd ed.