Bibliography

[AJ 2001], M. Ajtai, T. Jayram, R. Kumar, and D. Sivakumar. Counting inversions in a data stream. manuscript, 2001.

[AC 2003], Aslanyan L., Castellanos J., Mingo F., Sahakyan H., Ryazanov V., Algorithms for Data Flows, International Journal “Information Theories and Applications”, ISSN 1310-0513, 2003, Volume 10, Number 3, pp. 279-282.

[AM 1996], N. Alon, Y. Matias, and M. Szegedy. The space complexity of approximating the frequency moments. In Proc. of the 1996 Annual ACM Symp. on Theory of Computing, pages 20-29, 1996.

[GL 1980], E.N. Gordeev and V.K. Leontiev, Stability in problems of narrow places, JCM&MP, No. 4, Moscow, 1980.

[MM 2003], Manoukyan T and Mirzoyan V., Image Sequences and Bit-Plane Differences. “Computer Science & Information Technologies Conference”, Yerevan, September 22-26, pp. 284-286 (2003).

[GG 2002], A. Gilbert, S. Guha, P. Indyk, Y. Kotidis, S. Muthukrishnan, and M. Strauss. Fast, small-space algorithms for approximate histogram maintenance. In Proc. of the 2002 Annual ACM Symp. on Theory of Computing, 2002.

[V 1998], R.J. Vanderbei, Linear Programming; Foundations and Extensions, Kluwer Academic Publishers, Boston/London/Dordrecht, 1998.

Author's Information

Levon Aslanyan – Institute for Informatics and Automation Problems, NAS Armenia, P.Sevak St. 1, Yerevan-14, Armenia; e-mail: lasl@sci.am

MATRICIAL MODEL FOR THE STUDY OF LOWER BOUNDS

Jose Joaquin Erviti, Adriana Toni

Abstract: *Let V be an array. The range query problem concerns the design of data structures for implementing the following operations. The operation update(j,x) has the effect* $v_j \leftarrow v_j + x$, *and the query operation retrieve(i,j) returns the partial sum* $v_i + \ldots + v_j$. *These tasks are to be performed on-line. We define an algebraic model – based on the use of matrices – for the study of the problem. In this paper we establish as well a lower bound for the sum of the average complexity of both kinds of operations, and demonstrate that this lower bound is near optimal – in terms of asymptotic complexity.*

Keywords: zero-one matrices, lower bounds, matrix equations

ACM Classification Keywords: F.2.1 Numerical Algorithms and Problems

1 Introduction

Let V=(v_1 .. v_n) be an array of length n storing values from an arbitrary commutative semigroup S. We define the operations:

- retrieve(j,k): returns v_j+..+v_k $\quad \forall 1 \le j \le k \le n$
- update(j,x) : v_j := vj +x $\quad \forall 1 \le j \le n, \quad x \in S$ (1)

We refer to n as the size of the range query problem. We see that the complexity of executing an update(j,x) operation is constant meanwhile the worst complexity of a retrieve(i,j) operation is linear on n.

If we define the operations in a different way:

- retrieve(j,k) : returns v_k-v_{j-1} $\forall 1 \le j \le k \le n$ (consider v_0=0)
- update(j,x) : [forall $k \ge j$ do v_k:=v_k+x] $\forall 1 \le j \le n, \quad x \in S$ (2)

then the complexity of a retrieve(i,j) operation is constant meanwhile the worst case complexity of an update(j,x) operation is linear on n.

In both cases we are solving the same computational problem, that is to say: any sequence of update and retrieve operations over V produces the same results, no matter if the operations are implemented as in (1) or (2). With results we mean the outputs produced by the retrieve operations.

We can use different data structures involving a different number of variables storing values in the semigroup – others than the n variables organized as an array V - and provide the corresponding algorithms to implement the operations update and retrieve, and still be solving the same computational problem.

We work inside the semigroup model of computation: the array V can store values from an arbitrary commutative semigroup, and the implementations of the update and retrieve operations must perform correctly irrespective of the particular choice of the semigroup. In particular, the implementations are not permitted to utilize the substraction operation.

What kind of different data structures and programs are to be considered? We introduce right now the formal statement of the problem provided in [7].

We consider the class of data structures that involve program variables z_1, z_2 .. which store values in S. Stored in a variable z_i is the value $\sum_{j \in Y_i} v_j$ where Y_i is a specified subset of {1,2..n}. The query retrieve(j,k) is to be implemented with the program return $\sum_{i \in R_{jk}} z_i$ where R_{jk} is a specified set of integers. The update(j,x) operation is to be implemented with the program $z_l \leftarrow z_l$+x $\forall l \in U_j$, where $U_j = \{i \,/\, j \in Y_i\}$.

It is shown that a data structure in this class is correct if and only if

$$|U_l \cap R_{jk}| = \begin{cases} 1 & j \le l \le k \\ 0 & otherwise \end{cases}$$

In other words, we consider all the solutions consisting in a set of variables z_1, z_2 ... which store values in S, and such that a retrieve operation consists in adding up a subset of these variables, and an update(j,x) operation consists in incrementing by x a subset of these variables.

The complexity of the operation is defined as : complexity of update(j,x)=|U_j|, and complexity of retrieve(j,k) =|Rjk|.

From now on we will denote:

$$p = \frac{\sum_{j=1}^{n} U_j}{n} \quad \text{and} \quad t = \frac{\sum_{1 \le j \le k \le n} |R_{jk}|}{\binom{n}{2}}$$

that is to say, p represents the average complexity of the update operations – n is the number of such operations – and t represents the average complexity of the retrieve operations – in this case the number of operations is $\binom{n}{2}$.

The following results are discussed in this paper:

- In section 2 we define an algebraic model – based in the use of matrices – for the study of the problem.
- In section 3 we obtain a lower bound for p+t. We use a previously known result about the optimal lower bound for the average complexity of operations update and retrieve concerning the partial sums problem, which is a particular case of the range query problem in which the first argument of operation is fixed and equal 1.
- In section 4 we provide the corresponding amount p+t for some previously defined data structures solving our problem. This had not been calculated before, and it serves to verify that our lower bound is near optimal.

2 Matricial Model of Computation

An algebraic model of computation has been defined by M.L.Fredman (see [3]) for the study of the partial sums problem – a particular case of the range query problem in which the first argument of any operation is fixed and equal 1. It consists in defining the operations update and retrieve through pairs of zero-one matrices R, U verifying that their product is matrix T, where T is defined as

$$T=(t_{ij})_{i,j=1..n} \qquad \text{with} \quad t_{ij}=\begin{cases}1 & i \geq j\\ 0 & i \prec j\end{cases}$$

We establish in this section an algebraic model for the study of the more general problem known as the range query problem.

Our model includes all programs verifying:

- A set of variables $Z=\{z_1,z_2\ldots z_m\}$ which stores values in S is maintained, organized as an array.
- The operation retrieve(i,j) is executed adding up a subset of these variables.
- The operation update(j,x) is executed incrementing a subset of these variables by amounts which depend linearly on x.

Thus the model consists on triples <Z,R,U> where Z is the array, $R=(r_{ij})$ is a zero-one matrix of dimension $\frac{n(n+1)}{2}\times m$, and $U=(u_{ij})$ is a zero-one matrix of dimension $m\times n$ (m, the number of required program variables, may change although it has to be greater or equal n).

Associated with a triple there are the following programs to implement the update and retrieve operations.

<u>Definition 1</u>

If <Z,R,U> is a triple for the range query problem of size n, then the operations update and retrieve must be implemented through the following programs:

- *update(j,x) :* *for l:=1 to m do* $[z_l \leftarrow z_l + u_{lj}x]$
- *retrieve(i,j) :* *output* $\sum_{l=1}^{m} r_{kl} z_l$, *where* $k=\sum_{s=0}^{i-2}(n-s)+(j-i+1)$

Lemma 2 below establishes a condition on R,U which implies the correction of the programs given in Definition 1.

<u>Lemma 2</u>

Let H be the $\frac{n(n+1)}{2}\times n$ *dimensional matrix defined by:*

$$H_{ij}=\begin{cases}1 & l \le j \le l+(i-w_{l-1}-1)\\ 0 & otherwise\end{cases}$$

where

$w_k = \sum_{l=0}^{k-1}(n-l)$, $k=0...n$

$i \in \{(w_{l-1}+1)...w_l\}$, $l=1...n$

Then the programs in Definition 1 are correct if and only if RxU=H.

<u>Proof</u>

We find inspiration in the proof used in [3] (see Lemma 1 of [3]) to prove the correctness of the algorithms implementing the update and retrieve operations concerning the partial sums problem.

We have to consider the effect of the consecutive execution of operations [retrieve(i,j) ,update(r,x),retrieve(i,j)] for any i,j,r $\in \{1...n\}$, x $\in$ S.

Let q_1,q_2 be the output produced as a result of executing the two operations retrieve(i,j), the first and the second respectively.

From the definition of the operations given at the beginning of the Introduction section (see (1) in that section), it is obvious that the programs defined in Definition 1 are correct if and only if

$$q_2 - q_1 = \begin{cases}x & i \le r \le j\\ 0 & otherwise\end{cases}$$

But $q_1 = \sum_{l=1}^{m} r_{kl} z_l$, $q_2 = \sum_{l=1}^{m} r_{kl}(z_l + u_{lr}x)$ with $k = \sum_{s=0}^{i-2}(n-s) + (j-i+1) = w_{i-1} + (j-i+1),$

and so $q_2 - q_1 = \sum_{l=1}^{m}(r_{kl}u_{lr})x.$

By the definition of matrix H given in Lemma 2, and given that

$$\left[k = \sum_{s=0}^{i-2}(n-s) + (j-i+1)\right] \Rightarrow k \in \{(w_{i-1}+1)...w_i\},$$

we have

$$H_{kr} = \begin{cases}1 & i \le r \le i+(k-w_{i-1}-1)\\ 0 & otherwise\end{cases}$$

But $i+(k-w_{i-1}-1) = i+(j-i+1)-1 = j$, and the result may be deduced immediately.

<u>Remark 3</u>

Let us observe that if T^n *is the nxn dimensional matrix*

$T=(t_{ij})_{i,j=1..n}$ *with* $t_{ij}=\begin{cases}1 & i \ge j\\ 0 & i \prec j\end{cases}$

then we have that for all $k = 0...(n-1)$

$$H_{[w_k+1..w_{k+1}],[k+1..n]} = T^{n-k}$$

$$H_{[w_k+1..w_{k+1}],[1..k]} = 0$$

where $w_i = \sum_{s=0}^{i-1}(n-s)$ *for i=0..n,* $H_{[i..j],[r..s]} \equiv$ *the block of matrix H integrated by the rows belonging to the interval [i..j] and the columns belonging to the interval [r..s].*

So, for any size n of the range query problem, we may describe the corresponding matrix H^n through matrix T as

$$H = \begin{pmatrix} T^n \\ \overrightarrow{T^{n-1}} \\ \vdots \\ \overrightarrow{T^{n-(n-1)}} \end{pmatrix}$$

where

$$\overrightarrow{T^{n-i}} = \left(\begin{matrix} 0 & \dots & \dots & 0 \\ \vdots & & & \vdots \\ \vdots & & & \vdots \\ 0 & \dots & \dots & 0 \end{matrix} \middle| T^{n-i} \right) \qquad \text{i=1..(n-1)}$$

<u>Example 4</u>

Let us see the matrix H corresponding to the range query problems of size n=2 and n=4, which are named as H^2 and H^4 respectively. We have set the natural division in blocks off.

$$H^2 = \begin{pmatrix} 1 & 0 \\ 1 & 1 \\ \hline 0 & 1 \end{pmatrix}, \qquad H^4 = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 \\ \hline 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 \\ \hline 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 \\ \hline 0 & 0 & 0 & 1 \end{pmatrix}$$

Lemma 5 below establishes a result concerning the number of program variables $z_1..z_m$ required to implement any solution for the range query problem.

<u>Lemma 5</u>

Let R, U be zero-one matrices of dimensions nxm and mxn respectively such that RxU=H^n.

Then we have that $n \le m$.

<u>Proof</u>

First of all we prove that for any A, B zero-one matrices of dimensions nxm and mxn respectively such that AxB=T^n, we have that $n \le m$.

Let us proceed with this proof. We denote r, u, t the linear mappings associated with matrices A, B, T^n respectively. Then

$$R^n \xrightarrow{u} R^m \xrightarrow{r} R^n$$

From $T = A \circ B,$ it follows $n = \dim(\mathrm{Im}(t)) = \dim(r(u(R^n))) \le \dim(\mathrm{Im}(r)) \le m.$ So, necessarily $n \le m.$

Now let $R \times U = H^n$. Given that the product of the first n rows of R by matrix U is matrix T^n, we conclude trivially our result.

The following result connects the solutions for the range query problem inside our matricial model with the ones within the setting introduced in [7] that may be found also at the beginning of the Introduction section in this paper.

Remark 6

Given a triple <Z,R,U> which represents a solution for the range query problem of size n within our matricial model, where Z=(z_1..z_m), R=(r_{ij})$_{(n(n+1)/2)xm}$, U=(u_{ij})$_{mxn}$, and given $\{Y_1 \dots Y_m\}$, $\{R_{ij} \ / \ i, j = 1 \dots n\}$ defined as in Section 1, we have that

$$r_{ij} = \begin{cases} 1 & j \in R_{kl} \quad i = (l-k+1) + \sum_{s=0}^{k-2}(n-s) \\ 0 & otherwise \end{cases}$$

$$u_{ij} = \begin{cases} 1 & j \in Y_i \\ 0 & otherwise \end{cases}$$

Definition 7 below establishes the complexity of the operations within our model.

Defintion 7

Given the triple <Z,R,U> which represents a solution for the range query problem of size n within our matricial model, where Z=(z_1..z_m), R=(r_{ij})$_{(n(n+1)/2)xm}$ and U=(u_{ij})$_{mxn,}$, we define

- *Complexity of the operation retrieve(i,j):*

$$|\{r_{kl} \ / \ r_{kl} \neq 0 \wedge (1 \leq l \leq m)\}| \quad \textit{where} \quad k = (j-i+1) + \sum_{s=0}^{i-2}(n-s)$$

- *Complexity of the operation update(j,x):*

$$|\{u_{lj} \ / \ u_{lj} \neq 0 \wedge (1 \leq l \leq m)\}|$$

3 Lower Bound for the Sum of the Average Complexities

In this section we obtain a lower bound for the sum of the average complexity of the operations update and the average complexity of the operations retrieve which improves previously obtained lower bounds relating to this sum. We work inside our matricial model.

Let R, U be zero-one matrices representing a solution for the range query problem of size n, that is to say, such that RxU=H^n.

Assuming that m is the number of columns of matrix R and thus the number of rows of U, we have that the average complexity of the update operations is

$$p = \frac{\sum_{i=1}^{m}\sum_{j=1}^{n} u_{ij}}{n}$$

and the average complexity of the retrieve operation is

$$t = \frac{\sum_{i=1}^{n(n+1)/2} \sum_{j=1}^{m} r_{ij}}{n(n+1)\big/2}$$

In [7] has been established that for any data structure we have that $p + t = \Omega(\log n)$ (see Theorem 1 of [7]), and more exactly, the result says that

$$(2n^2/9)\log_e n + O(n^2) \le 2n^2 p + 2\frac{n(n+1)}{2}t$$

(from this inequality a lower bound for p+t may be obtained with little calculation).

We provide a different method which let us obtain easily a lower bound for $2n^2 p + n(n+1)t$ by reusing the knowledge of an optimal lower bound for the average complexity of the operations update and retrieve (all of them being considered together in this case) with regard to the partial sums problem of size n.

It has been proved (see [6]) that for $2^k \le n < 2^{k+1}$, we have that if AxB=T^n, being A and B nxm and mxn dimensional matrices respectively, then

$$\sum_{i=1}^{n}\sum_{j=1}^{m} a_{ij} + \sum_{i=1}^{m}\sum_{j=1}^{n} b_{ij} \ge n[\log_2 n] + [\log_2 n] + (m - 2^k) + 2$$

We proceed to obtain a lower bound for $2n^2 p + n(n+1)t$. It must be observed that we will reach a lower bound I which the constant factor affecting the term which leads the expression is greater that the one in the lower bound obtained in [7]: ours is $n^2 \log_2 n$, meanwhile the one in [7] is $n^2\left(\frac{2}{9}\log_e n\right)$. In particular, $\log_2 n > 5\left(\frac{2}{9}\log_e n\right)$.

<u>Theorem 8</u>

Let R, U be zero-one matrices such that RxU=H^n.

Then we have that

$$2n^2 p + n(n+1)t \ge n^2(\log_2 n - \frac{1}{2}\log_2 e) + 2n\log_2 n + 2n + \frac{10}{4}\log_2 e$$

where

$$p = \frac{\sum_{i=1}^{m}\sum_{j=1}^{n} u_{ij}}{n} \quad , \quad t = \frac{\sum_{i=1}^{n(n+1)/2}\sum_{j=1}^{m} r_{ij}}{n(n+1)\big/2}$$

and m represents the number of columns in matriz R and tus the number of rows in U.

<u>Proof</u>

We have that

$$R \times U = H^n = \begin{pmatrix} T^n \\ \overrightarrow{} \\ T^{n-1} \\ \vdots \\ \overrightarrow{} \\ T^{n-(n-1)} \end{pmatrix}$$

where

$$\overrightarrow{T^{n-i}} = \left(\begin{matrix} 0 & \dots & \dots & 0 \\ \vdots & & & \vdots \\ \vdots & & & \vdots \\ 0 & \dots & \dots & 0 \end{matrix} \middle| T^{n-i} \right) \qquad \text{i=1..(n-1)}$$

Let $R_{[i\dots j],[1\dots m]}$ be the block consisting of the rows of R belonging the interval [i..j], for $1 \le i \le j \le n(n+1)/2.$

We consider matrix R as divided in n blocks, R1..Rn, where R1 consists of the first n rows of R, R2 of the following (n-1) rows…and being the last block Rn formed exclusively by the last row of R. That is to say, if we call $z_i = \sum_{j=0}^{i-1}(n-j), \quad i = 0\dots n,$ then $R_i = R_{z_{i-1}+1\dots z_i,1\dots m}.$

Using this notation we have that $R_i \times U = T^{n-i+1}.$

We apply the lower bound that we already know for the partial sums problem to every pair Ri , U with $i = 1\dots n$ (see [6]). Let us observe that $|R| + n|U| = \binom{n}{2}t + n^2 p$ where |R|, |U| denotes the number of non-zero entries in matrix R and U respectively .

We have

$$|R_i| + |U| = \sum_{l=z_{i-1}+1}^{z_i}\sum_{s=1}^{m} r_{ls} + \sum_{l=1}^{m}\sum_{s=1}^{n} u_{ls} \ge (n-i+1)log_2(n-i+1) + log_2(n-i+1) + (m-2^k) + 2$$

So, and without taking account of the factor $(m-2^k)$,

$$|R| + n|U| = \sum_{i=1}^{n(n+1)/2}\sum_{j=1}^{m} r_{ij} + n\sum_{i=1}^{m}\sum_{j=1}^{n} u_{ij} \ge$$

$$\ge [(n+1)\log_2 n + 2] + [n\log_2(n-1) + 2] + \dots + [(2+1)\log_2 2 + 2] + 2 =$$

$$= [(n+1)\log_2 n + n\log_2(n-1) + \dots + (2+1)\log_2 2] + 2n$$

Let us call $\delta(n) = (n+1)\log_2 n + \dots + (2+1)\log_2 2$ and then we have

$$|R| + n|U| \ge \delta(n) + 2n$$

We observe that

$$\sum_{i=1}^{n(n+1)/2}\sum_{j=1}^{m} r_{ij} + n\sum_{i=1}^{m}\sum_{j=1}^{n} u_{ij} = \frac{n(n+1)}{2}t + n^2 p,$$

and we are trying to establish a lower bound for $2n^2 p + n(n+1)t.$

Let $f(x) = (x+1)\log_2 x.$ Then

$$\int_1^n f(x)dx \le \delta(n) \le \int_2^{n+1} f(x)dx$$

Considering $\log_2 x = \log_2 e \log_e x,$ we integrate the expression

$$\log_2 e \int (x+1)\log_e x dx$$

so if we call

$$u = \log_e x \Rightarrow du = dx / x$$
$$dv = (x+1)dx \Rightarrow v = (x+1)^2 / 2$$

then

$$\int (x+1)\log_e x dx = \frac{(x+1)^2}{2}\log_e x - \frac{1}{4}x^2 - x - \frac{1}{2}\log_e x$$

(the constant factor log$_2$e will be taken into account later)

So

$$\int_1^n f(x)dx = \frac{(n+1)^2}{2}\log_e n - \frac{1}{4}n^2 + \frac{5}{4} - n - \frac{1}{2}\log_e n$$

Then

$$n^2 p + \frac{n(n+1)}{2}t \geq \delta(n) + 2n \geq$$

$$\geq 2n + \log_2 e\left[\frac{(n+1)^2}{2}\log_e n - \frac{1}{4}n^2 + \frac{5}{4} - n - \frac{1}{2}\log_e n\right] \geq$$

$$\geq \frac{n^2}{2}\log_2 n + n\log_2 n + n - \frac{1}{4}n^2\log_2 e + \frac{5}{4}\log_2 e$$

Multiplying by 2 both sides of the inequality we obtain the result stated in this theorem, that is

$$2n^2 p + n(n+1)t \geq n^2(\log_2 n - \frac{1}{2}\log_2 e) + 2n\log_2 n + 2n + \frac{10}{4}\log_2 e\,.$$

Remark 9

Let us observe that from this result a lower bound of $\Omega(\log n)$ *for* $p+t$ *can be deduced: dividing both sides of the final inequality by* $2n^2$ *we obtain* $p + \left(\frac{1}{2} + \frac{1}{2n}\right)t \geq \log_4 n + (\log_2 n)/n + 1/n - (\log_2 e)/4 + (10\log_2 e)/8n^2,$ *and so, given that* $\frac{1}{2} + \frac{1}{2n} \leq 1 \quad \forall n \geq 1,$ *we conclude that* $p + t \in \Omega(\log n.)$

4 Complexity of Certain Data Structures

In this section we establish the value of $p+t$ for some previously defined data structures solving our problem – more precisely , we provide the amount $\binom{n}{2}t + n^2 p$. In particular, trees have been defined in [7] (see Theorem 2 in that paper) verifying that given a fixed integer k, the worst case complexity of an update operation is k , and the worst case complexity of a retrieve is $O(n^{\frac{1}{k-1}})$. The definition of the trees satisfying these conditions is straightforward, based on the use of an $n^{1/(k-1)}$-ary tree of height k with n leaves. An update(j,x) operation has to be executed incrementing by x the value stored in the j-th leaf from left to right and in all the nodes belonging to the path from this leaf to the root, and a retrieve(j,x) operation has to be executed adding up the values in the minimum set of nodes verifying that the union of its successors includes exactly the leaves from i to j, and the intersection of its successors is disjoint pairwise - the idea is that each internal node stores the sum of the values in its sons.

Let $T_{r^{h-1}}$ denote the r-ary tree of deep h with $n=r^{h-1}$ leaves. We claim

Theorem 10

Given $T_{r^{h-1}}$, *we have that*

$$\binom{n}{2}t + n^2 p = (r-1)(h-1)\left(n + \frac{n^2}{2}\right) - \frac{n(n-1)(r+1)}{3} + 1 + n^2 h$$

Proof

The proof involves heavy calculation and will not be included in this paper.

If we compare the result stated in Theorem 10 with our lower bound for $\binom{n}{2}t + n^2 p$ we obtain the following corollary.

Corollary 11

Given $T_{r^{h-1}}$, *in order to minimize the amount* $\binom{n}{2}t + n^2 p$ *we must choose* $r=2$, $h=\log_2 n+1$, *being* $n=r^{h-1}$.

In this case we have that $\binom{n}{2}t + n^2 p = 3n^2 \log_4 n - \Theta(n^2)$. *In Theorem 8 we proved that* $\binom{n}{2}t + n^2 p \geq n^2 \log_4 n - \Theta(n^2)$, *so this lower bound is near optimal. Let us observe that this means that the* $\Omega(\log n)$ *lower bound for* $p+t$ *is optimal in terms of asymptotic complexity.*

5 Bibliography

[1] W.A. Burkhard, M.L. Fredman, D.J.Kleitman, *Inherent complexity trade-offs for range query problems*, Theoretical Computer science, North Holland Publishing Company 16, (1981) pp.279--290.

[2] M.L. Fredman, *Lower Bounds on the Complexity of some Optimal Data Structures*, Siam J. Comput., Vol.10, No.1 (1981) pp.1--10.

[3] M.L. Fredman, *The Complexity of Maintaining an Array and Computing its Partial Sums*, J.ACM, Vol.29, No.1 (1982) pp.250--260.

[4] M.L. Fredman, H.Hampapuram, *Optimal Bi-Weighted Binary Trees and the Complexity of Maintaining Partial Sums*, Siam J. Comput., Vol.28, No.1 (1998) pp.1--9.

[5] M.L. Fredman, M.E. Saks, *The Cell Probe Complexity of Dynamic Data Structures*, Proceedings of 21 st ACM Symposium on Theory of Computing, (1989) pp.345--354.

[6] A. Toni, *Lower Bounds on Zero-one Matrices*, Linear Algebra and its Applications, 376 (2004) 275--282.

[7] D.J. Volper, M.L. Fredman, *Query Time Versus Redundancy Trade-offs for Range Queries*, Journal of Computer and System Sciences 23, (1981) pp.355--365.

[8] A.C. Yao, *On the Complexity of Maintaining Partial Sums*, Siam J. Comput., Vol.14, No.2 (1985).

Authors' Information

Jose Joaquin Erviti – jerviti@fi.upm.es

Adriana Toni – atoni@fi.upm.es

Facultad de Informática, Universidad Politécnica de Madrid, Spain