Optimal Skewed Allocation on Multiple Channels for Broadcast in Smart Cities

Giorgio Audrito
Department of Computer Science,
University of Pisa, Largo B. Pontecorvo,
56100 Pisa, Italy.
Email: giorgio.audrito@gmail.com

Daniele Diodati and Cristina M. Pinotti
Department of Computer Science and Mathematics,
University of Perugia, Via Vanvitelli, 1,
06123 Perugia, Italy
Email: daniele.diodati@dmi.unipg.it, cristina.pinotti@unipg.it

***Abstract*—We consider the problem of allocating N uniform data to K transmission channels so as the Average Expected Delay (AED) is minimized. This problem arises in designing efficient data-diffusion broadcast algorithms in a smart environment. We show that the basic dynamic programming algorithm for solving the uniform allocation problem can be speedup up to $O(NK)$ time by applying an optimal algorithm to find the row-minima of totally monotone matrices. Such a new algorithm is always faster than the best previously known algorithm for the uniform allocation problem that runs in $O(NK \log N)$. Moreover, it is computationally optimal for the uniform allocation of up to N data and K channels. We then reduce the largest allocation problem, i.e., the subproblem with exactly N data and K channels, to the problem of finding a minimum weight K-link path in a particular directed acyclic graph. We also present two heuristics and we show by extended simulations their effectiveness in practical scenarios. Both the K-link path algorithm and the heuristics are much faster than $O(NK)$. We then compare the behaviours of our algorithms on the online version of the allocation problem in which new single items are inserted for broadcast.**

***Index Terms*—Data broadcast, multiple channels, average expected delay, Monge matrix, dynamic programming**

I. Introduction

Data broadcast on multiple channels is a popular data dissemination method to distribute a set of data [9], [11], [12]. Specifically, the data broadcast system that we imagine works as follows. Users require informations in which they are interested to a data service. Then the users wait –either listening to radio channels or watching to screens – until the required info will be broadcast. The server of the data service collects the requests and, based on their popularity, it decides a data schedule to be broadcast (that is, it decides for each data in which order, on which channel/screen, how frequently it is broadcast). Broadcasting a single item, the server serve all the users that have done the same request. The level of satisfaction of the users will be high if the experienced waiting time is short.

Because of its scalability and flexibility, a multiple wireless data broadcast system is especially suitable for information of public interest, such as weather, traffic, commercial advertisements and stock quote. We can imagine to broadcast public information shared on multiple large displays in open areas of smart-cities. For example, malls could list the commercial offers of the day, based on their popularities, over multiple displays. Or, alarms, weather and traffic infos could be distributed, based on their pressure, on wireless screens or on wireless radio channels along the highways. This algorithm could be used to distribute flight and train schedule in airports or stations, especially with regards to delayed trains or flights, for which a FIFO order based on the departure time is no longer applicable. In each of these examples, it is also quite reasonable that the info are formatted in a regular way and they have all the same size. In general, the radio-channel, screen, display where data will be distributed is denoted in this paper as channel. So, we imagine a base station or a server will repeatedly broadcast a set of data over multiple channels according to a predefined schedule. A client, e.g., an application on a smartphone or a user, will listen to the channels until the requested data is broadcast in order to download it. An efficient allocation schedule on multiple wireless channels wants to minimize the *Average Expected Delay* (AED), that is, the average amount of time (or energy) that the client waits listening to the channels before receiving the data.

In [7], the problem of minimizing the AED has been solved assuming: *i*) *uniform* length of items, that is, every item requires one time slot to be broadcast on any channel; *ii*) *skewed* data allocation for channels, that is, a different number of items is assigned to each channel; *iii*) *flat* data scheduling for each single channel, that is, the items assigned to each channel are repeated in a cyclic carousel.

For such a *K-Uniform Allocation* problem, also called the *AED* problem, the best algorithm known so far was devised in [7] and runs in $O(NK \log N)$ time. In [7], it was left as an open problem to decide whether the time complexity of the K-uniform allocation problem could be further improved or not. In this work, we present new algorithms for the K-uniform allocation problem (and for some variants of it) that are faster than the one proposed in [7] and that can be proved optimal for the time complexity or the quality of the solution.

The rest of this paper is so organized. Section I-A introduces

Work supported by the Research Grant 2010N5K7EB PRIN 2010 ARS Techno-Media (Algoritmica per le Reti Sociali Tecno-mediate) from the Italian Ministry of University and Research.

978-1-5090-0898-8/16/$31.00 ©2016 IEEE

the problem statements and gives some notations and definitions. Section II reviews the currently known algorithm solving the AED problem and presents two new algorithms for the AED-full and the AED-single problems that run, respectively, in $O(NK)$ and $O(N \log N)$ time thus improving on the previous solutions which required $O(NK \log N)$ time. Section III presents an ϵ-approximate solution for the *continuous* version of the AED problem that runs in $O(K \log^2 \frac{N}{\epsilon})$ time and it shows how to convert such a solution for the single-AED (discrete) problem. The effectiveness of the converted solution is experimentally tested on data that follow the Zipf distribution, a benchmark distribution for the demand probabilities [5], [13]. In Section IV, the online AED problem is studied. Finally, conclusions are offered in Section V.

A. Problem statements

Let K be the number of wireless channels available and N be the number of items $\{d_1, d_2, \ldots, d_N\}$ to be broadcast, with associated demand probabilities $\{p_1, p_2, \ldots, p_N\}$ where p_i represents how frequently item d_i is requested. Each item d_i, $1 \leq i \leq N$ has uniform *length*, that is, it requires one time slot to be broadcast on any channel. In the following, we shall consider the following instances of the K-uniform allocation problem:

AED: The goal is to find the partition of the N items into K groups $G_1, \ldots, G_K$ that minimizes the *average expected delay* AED, which is defined as follows.

Group G_j collects the data items assigned to channel j, with $1 \leq j \leq K$. The cardinality of G_j is N_j. Since the items in G_j are cyclically broadcast according to a flat schedule and since we assume that each data item has unit length, N_j is the schedule period of channel j. The *expected delay* for receiving item $d_i \in G_j$ is $\frac{N_j}{2}$, that is, half of the period of channel j because the clients can start to listen, with the same probability, at any instant of the schedule period. Then:

$$\text{AED} = \sum_{i=1}^{N} \frac{N_j}{2} p_i = \frac{1}{2} \sum_{j=1}^{K} \left(N_j \sum_{d_i \in G_j} p_i \right) \tag{1}$$

Notice that the AED can be split into the sum of the per-channel average waiting time defined as: $w_{\text{AED}}(G_j) = \frac{N_j}{2} \sum_{d_i \in G_j} p_i$.

Single and full versions: We shall consider the following variations and denote them by postponing a "-single", "-full" suffix to the AED acronym.

- *single*: the requirement is to solve only the largest subproblem with $n = N$ and $k = K$;
- *full*: the requirement is to solve *all* the subproblems with $1 \leq n \leq N$, $1 \leq k \leq K$ (a prefix of the items and of the channels);

II. The AED problem

The AED-full problem has been first introduced in [11] and further developed in [7]. They prove that optimal solutions for the AED problem have to be sought among the *segmentations*, that is, the partitions that preserve the probability order, e.g. $p_1 \leq p_2 \ldots \leq p_N$. An N-segmentation is a partition $G_1, \ldots, G_K$, such that if $d_i \in G_j$ and $d_k \in G_j$, then $d_h \in G_j$ whenever $i \leq h \leq k$. That is, each group is made of consecutive sorted items. Based on that, the cost of group $G_i^j = \{i+1, \ldots, j\}$ is $w(i,j) = \frac{j-i}{2} \sum_{k=i+1}^{j} p_k$ and the AED-full problem can be solved via the following dynamic programming for $1 \leq k \leq K$, $k \leq n \leq N$:

$$opt(k,n) = \begin{cases} w(0,n) & \text{if } k = 1 \\ \min\limits_{k-1 \leq \ell \leq n-1} \{opt(k-1,\ell) + w(\ell,n)\} & \text{if } k \geq 2 \end{cases} \tag{2}$$

where $opt(k,n)$ denotes the cost of the optimal solution $OPT(k,n)$ for the k-uniform allocation problem applied to data items $d_1, \ldots d_n$. Moreover, the *final border* $F[k,n]$ of $OPT(k,n)$, that is, the index of the last item that belongs to group G_{k-1} of $OPT(k,n)$, is:

$$F[k,n] = \arg \min_{k-1 \leq \ell \leq n-1} \{opt(k-1,\ell) + w(\ell,n)\} \tag{3}$$

Note that, in case of multiple indices that lead to the same $opt(k,n)$ value, $F[k,n]$ is set to the minimum index.

Clearly, the cost of the K-uniform allocation problem applied to N data items can be found in $opt(K,N)$ and the solution $OPT(K,N)$ can be built backwards from the final border $F[K,N]$. The dynamic programming implementation stores in matrices $\mathbb{M}$ and $\mathbb{F}$ the optimal costs $opt(k,n)$ and the final borders $F[k,n]$, respectively. The matrices $\mathbb{M}$ and $\mathbb{F}$ are filled row-by-row for increasing values of k. The algorithm takes overall $O(N^2K)$ time [11]. Indeed, $M[k,n] = opt(k,n)$ is the minimum of $O(n)$ values, each of which can be computed in $O(1)$ time because $w(\ell,n) = \frac{n-\ell}{2}(P(n) - P(\ell))$ where $P(n) = \sum_{i=1}^{n} p_i$, $1 \leq n \leq N$. The prefix sum vector P can be computed just once in an $O(N)$ time preprocessing step.

Ardizzoni et al. proposed the *Dichotomic* algorithm, which takes advantage from the fact that the final border of the optimal solution moves only towards right when the set of data items increases [7]. This implies that if the borders $F[k,l]$ and $F[k,r]$ are known for some $1 \leq l \leq r \leq N$, then $F[k,c]$ has to be sought only between $F[k,l]$ and $F[k,r]$, for any $l \leq c \leq r$. Thus, the recurrence in Equation 2 can be rewritten as:

$$opt(k,c) = \begin{cases} w(0,n) & \text{if } k = 1 \\ \min\limits_{\ell \in \{F[k,l] \ldots F[k,r]\}} \{opt(k-1,\ell) + w(\ell,c)\} & \text{if } k \geq 2 \end{cases}$$

By this property, the evaluation via dynamic programming of the entries of $\mathbb{M}$ can be accelerated. Namely, for a fixed value of k, all the values $opt(k,n)$ in the k-th row of $\mathbb{M}$ can be computed in $\Theta(\log N)$ phases, filling in each phase the middle points of the entries filled in the previous phases and just checking overall $O(N)$ possible borders in each phase. In this way, the Dichotomic algorithm takes $O(N \log N)$ time to fill one row of $\mathbb{M}$ and $O(NK \log N)$ to solve the K-uniform allocation problem.

From now on, we assume that the data items are given sorted, and thus we shall derive the time complexity of the algorithms without taking into account the sorting step. We remark that all the complexities previously given for algorithms known in literature ($O(N^2K)$ in [11], $O(NK\log N)$ in [7]) are also derived without taking into account the sorting step; which is in any case needed and whose time complexity $O(N\log N)$ is negligible with respect to the complexities of all algorithms presented when $K=\Omega(\log N)$ (except for the heuristics which run in $O(K\log^2 N)$).

A. The new $O(NK)$ algorithm for the AED-full problem

Definition 1: A 2×2 matrix $\begin{bmatrix} a & b \\ c & d \end{bmatrix}$ is *concave Monge* if $a+d \le b+c$. An $m\times n$ matrix $\mathbb{A}$ is concave Monge if every 2×2 submatrix is concave Monge. That is, for all $1\le i<m$ and $1\le j<n$,

$$A[i,j]+A[i+1,j+1] \le A[i+1,j]+A[i,j+1]$$

or, equivalently:

$$A[i,j+1]-A[i,j] \ge A[i+1,j+1]-A[i+1,j].$$

Definition 2: A 2×2 matrix is *monotone* if the minimum of the upper row is not to the right of the minimum of the lower row. More formally, $\begin{bmatrix} a & b \\ c & d \end{bmatrix}$ is *monotone* if $b<a$ implies that $d<c$ and $b=a$ implies that $d\le c$. An $m\times n$ matrix $\mathbb{A}$ is *totally monotone* if every 2×2 submatrix of $\mathbb{A}$ is monotone. It is well known (see [1], [4], [6]):

Fact 1: Every concave Monge matrix is totally monotone.

For a totally monotone matrix, we mention the following important result:

Fact 2: The minimum in each row of a totally monotone matrix of size $n\times m$ can be computed in $O(n+m)$ time by applying the SMAWK algorithm proposed in [1].

The rest of this section explains how to apply the SMAWK algorithm to optimally solve the AED-full problem (a short description of the SMAWK algorithm is given in Appendix).

Define the upper triangular *single-channel cost* matrix $\mathbb{W}$ of size $N\times N$ as $W[i,j]=w(i,j)$ for $0\le i<j\le N$, where $w(i,j)=\frac{j-i}{2}\sum_{k=i+1}^{j} p_k$ is defined as the cost of the group $G_i^j=\{i+1,\ldots,j\}$.

Lemma 1: The single-channel cost (upper triangular) matrix $\mathbb{W}$ is a concave Monge matrix, for $0\le \ell<n<N$:

$$w(\ell,n)+w(\ell+1,n+1) \le w(\ell,n+1)+w(\ell+1,n).$$

Proof: The simplest way to prove this result is to observe that the column-marginal is not-increasing. In fact, for $0\le \ell < n \le N$,

$$w(\ell,n+1)-w(\ell,n)=\frac{n-\ell+1}{2}p_{n+1}+\frac{1}{2}\sum_{q=\ell+1}^{n} p_q \ge$$

$$\frac{n-\ell}{2}p_{n+1}+\frac{1}{2}\sum_{q=\ell+2}^{n} p_q = w(\ell+1,n+1)-w(\ell+1,n).$$

■

It is worthy to point out that the Monge property for the single-channel costs holds independently of the order of the items.

If we fill the lower triangular matrix $\mathbb{W}$ with $+\infty$ values, the matrix $\mathbb{W}$ is totally monotone.

From now on let us denote $S^k_{\ell,n}=opt(k-1,\ell)+w(\ell,n)$. Thus, $S^k_{\ell,n}$ can be seen as a single entry of a three-dimensional matrix $\mathbb{S}$, with the first dimension k varying on the set of channels $[1,\ldots,K]$, the second dimension ℓ on the possible positions of the final border of the group G_{k-1} (i.e., $k-1\le \ell\le N-1$), and the third dimension n on the set of data items $[1,\ldots,N]$.

After having fixed the first dimension of $\mathbb{S}$ to the value k, we may extract from $\mathbb{S}$ the two dimensional matrix $\mathbb{S}^k=[S^k_{\ell,n}]$, $k-1\le\ell\le N-1, k\le n\le N$. Thus, the three dimensional $\mathbb{S}=\cup_{k=1}^{K}\mathbb{S}^k$, that is, $\mathbb{S}$ can be decomposed into K two dimensional matrices, one for each value of k. Accordingly to the new notation, for $k\ge 2$, we can rewrite Eqs. 2 and 3 as:

$$opt(k,n)=\min_{k-1\le\ell\le n-1}\{S^k_{\ell,n}\}; \tag{4}$$

$$F[k,n]=\arg\min_{k-1\le\ell\le n-1}\{S^k_{\ell,n}\} \tag{5}$$

Then, $opt(k,n)$ is the minimum in row n of $\mathbb{S}^k$.

Now we prove that each matrix $\mathbb{S}^k$, for $1\le k\le K$, is totally monotone. Namely, if we add the same value $opt(k-1,\ell)+opt(k-1,\ell+1)$ to both sides of the concave Monge condition for the single-channel cost matrix $\mathbb{W}$, we obtain the concave Monge condition for matrix $\mathbb{S}^k$:

$$\begin{aligned} &\underbrace{opt(k-1,\ell)+w(\ell,n)}_{S^k_{\ell,n}}+\underbrace{opt(k-1,\ell+1)+w(\ell+1,n+1)}_{S^k_{\ell+1,n+1}} \\ &\le \underbrace{opt(k-1,\ell)+w(\ell,n+1)}_{S^k_{\ell,n+1}}+\underbrace{opt(k-1,\ell+1)+w(\ell+1,n)}_{S^k_{\ell+1,n}} \end{aligned} \tag{6}$$

Hence, to compute the row k of the matrix $\mathbb{M}$ of the classical dynamic programming algorithm, it is sufficient to apply the SMAWK algorithm to matrix $\mathbb{S}^k$. Specifically, by applying Fact 2, it holds:

Lemma 2: Fixed any $k\ge 2$, the SMAWK algorithm can compute the values $opt(k,n)$ for $k\le n\le N$ in $O(N)$ time, if the values $opt(k-1,n)$ for $k-1\le n\le N$ are known (that is, if the row $k-1$ of the matrix $\mathbb{M}$ is known).

Proof: To apply the SMAWK algorithm to the totally monotone matrix $\mathbb{S}^k$ it is required that each entry $S^k_{\ell,n}$ can be computed in constant time. This trivially holds since we assume to know $opt(k-1,n)$ for $1\le n\le N$ and since we have shown that each entry of the single-channel cost matrix $\mathbb{W}$ can be computed in constant time. ■

Theorem 1: The AED-full problem can be solved in $O(NK)$ time by applying $K-1$ times the SMAWK algorithm [1] as illustrated in Algorithm *SMAWK-AED*. The complexity of such an algorithm is optimal since it solves NK subproblems.

Clearly, the SMAWK-AED algorithm is always faster than the Dichotomic algorithm.

Finally, observe that the SMAWK-AED algorithm can be adapted to solve the AED-single problem. Indeed, a careful

Algorithm 1: SMAWK-AED

Input: $\mathbb{W}$, N, K
Output: $opt(k,n)$, $1 \le k \le K$, $1 \le n \le N$

$opt(1,n) = w(0,n)$, $1 \le n \le N$;
for $2 \le k \le K$ **do**
  $opt(k,n) = +\infty$ for $1 \le n \le k-1$;
  $opt(k,k), \ldots, opt(k,N) \leftarrow$ SMAWK($\mathbb{S}^k$)

analysis of the size of the matrices involved in the $K-1$ repeated applications of the SMAWK algorithm required to compute $opt(K,N)$ leads to the following result:

Theorem 2: For fixed $K \ge 2$ and $N \ge K$, AED-single can be solved in $O(K(N-K))$ time by solving $K-1$ instances of the SMAWK algorithm. ■

Observe that for $K = \theta(1)$ and $K = N - \theta(1)$, AED-single can be solved in $O(N)$ time.

In the next section we show that faster solutions can be found for AED-single if we abandon the dynamic programming technique.

B. The AED-single problem: Minimum Weight K-Link Path

In order to outperform the $O(NK)$ time of SMAWK-AED we shall need to drop the requirement of solving all subproblems of allocating a prefix of the items $\{1,\ldots,n\}$ to k channels with $1 \le n \le N$, $1 \le k \le K$, and instead concentrate on solving only the largest subproblem with N data items and K channels. The largest subproblem can be considered the most meaningful in practice, but we will show that the ad-hoc solution for the largest subproblem lacks of flexibility and cannot be easily adapted to variants of the original problem, like those proposed in this work.

AED-single can be reduced to the problem of finding a minimum weight K-link path (see [2], [3], [10]) in a particular *directed Acyclic Graph* (DAG) G. This problem asks to identify a path from v_0 to v_N in G consisting of exactly K edges whose cost is minimum among all such paths.

In our solution, the graph G has a vertex for each data item d_i, with $1 \le i \le N$, plus a dummy starting vertex v_0 that marks the start. For any pair of positions ℓ and n such that $0 \le \ell \le N$, we have an edge (v_ℓ, v_n) whose cost is the single-channel cost $w(\ell, n)$.

Very efficient solutions for the problem of computing a minimum weight K- link path have been provided in literature (see [2], [3], [10]) whenever the DAG satisfies the concave Monge condition. Let us borrow from [8] the survey of the best known solutions for such a problem.

Fact 3 ([8, Thm. 1]): Given a DAG G satisfying the concave Monge condition and whose weights are integers, and given $K \le N$, a minimum weight K-link path in G can be computed in $O(N \log U)$ time where U is the maximum absolute value of the weights.

In fact, this algorithm performs a binary search for an additive constant d to be added to every edge of G, in order to ensure that the Single Source Shortest Path (SSSP) is made of exactly K edges. Since adding $d = -U$ grants that the SSSP is made of N links, and adding $d = U$ forces the SSSP to be made of 1 link only, this strategy performs $\log U$ instances of SSSP which are solved in $O(N)$ time by applying SMAWK algorithm.

Thus Fact 3 provides a weakly polynomial algorithm for the problem. In fact, the demand probabilities $p_1, p_2, \ldots, p_N$ of data items are typically frequencies derived by observing requests in a request-log of total length, say, L. Thus, we can label the edges of G with integral weights by appropriately multiplying each of these frequencies by L. In this way, the factor $\log U$ in the time complexity of Fact 3 is $O(\log N + \log L)$ for the AED problem. For completeness, we notice that there exists also a solution whose time complexity is independent on the values of the weights.

Fact 4 ([8, Thm. 2]): Given a DAG G satisfying the concave Monge condition, and given $K = \Omega(\log N)$, a minimum weight K-link path in G can be computed in $O(NK^\epsilon)$ time for any fixed ϵ.

III. Continuous allocation problem

In this section we shall consider the *continuous version* of AED-single, and show that an ϵ-approximated solution for it can be found in $O(K \log^2(N/\epsilon))$ time. Such a solution can be converted into a solution for the discrete AED-single either directly by rounding, or by applying a "hinted" version of the dynamic programming algorithm running in $O(K \log^2(N))$ time. Experimental tests show that this approach is effective both in time and accuracy on practical data sets.

In the remainder of this section, we refer to the AED function defined in Eq. 1 as the *discrete* AED.

Let $\mathbf{p} : [0, N] \to \mathbb{R}^+$ be an integrable function. Define the *continuous channel cost* $w(x,y)$ with $x \le y$ as

$$w(x,y) = \frac{y-x}{2} \int_x^y \mathbf{p}(z)\mathrm{d}z$$

The continuous AED-single asks to find $K-1$ real numbers F_i, $i = 1 \ldots K-1$ such that $0 = F_0 < F_1 \ldots < F_{K-1} < F_K = N$ and

$$\text{cAED} = \sum_{i=1}^{K} w(F_{i-1}, F_i) \tag{7}$$

be minimized. We say that $F_i : \ i \le K$ is an ϵ-solution if $|F_i - G_i| < \epsilon$ for all $i \le K$ where $G_i : \ i \le K$ is a real solution of continuous AED-single.

Notice that in case $\mathbf{p}$ is a step-function following the sequence p_i (i.e. $\mathbf{p}(x) = p_{\lfloor x+1 \rfloor}$), the continuous channel cost coincides with the discrete channel cost associated to the sequence for all integer x, y. We could also choose $\mathbf{p}$ so as to interpolate smoothly the sequence p_i, for example, the linear interpolation $\mathbf{p}(i - \frac{1}{2} + x) = (1-x)p_i + xp_{x+1}$ for $x \in [0,1]$. However, this is not as effective as the step-function in practice.

As in the discrete case, we are able to prove that (a continuous version of) the concave Monge property holds for w.

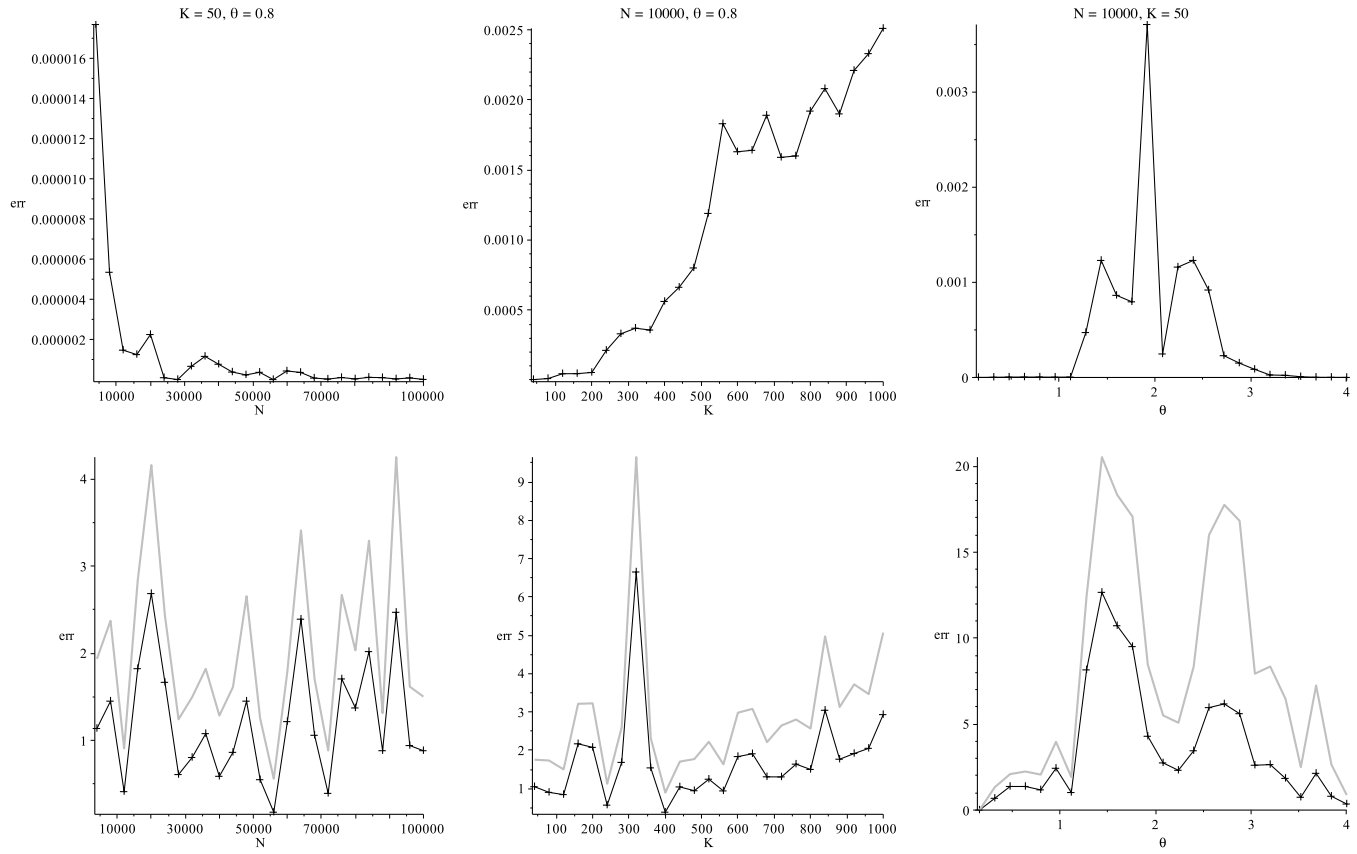

Fig. 1: AED relative error (top) and border position mean error with standard deviation added in grey (bottom) for heuristic `cont-round`. All tested input cases are exactly solved by heuristic `cont-dp` and thus are not reported.

Lemma 3: The function $w(x, y)$ satisfies the concave Monge property, that is, given $0 \leq x_1 < x_2 \leq y_1 < y_2 \leq N$,

$$w(x_1, y_1) + w(x_2, y_2) < w(x_2, y_1) + w(x_1, y_2).$$

Proof: We prove that

$$w(x_1, y_1) - w(x_2, y_1) < w(x_1, y_2) - w(x_2, y_2).$$

In fact,

$$w(x_1, y) - w(x_2, y) = \frac{y - x_1}{2} \int_{x_1}^{x_2} \mathbf{p}(z)\mathrm{d}z + \frac{x_2 - x_1}{2} \int_{x_2}^{y} \mathbf{p}(z)\mathrm{d}z$$

which is strictly increasing as y increases. ∎

In analogy with the discrete case, we can then prove that for any fixed K, F_i for all $i \leq K$ monotonically increase as N increase. If we allow N to range over the reals, an easy continuity argument shows that F_i increase *continuously* for all $i \leq K$. Furthermore, we can exploit the continuity and use the partial derivatives $w_1(x, y)$ and $w_2(x, y)$, that is,

$$\begin{aligned} w_1(x, y) &= \frac{\partial w(x, y)}{\partial x} \\ &= -\frac{1}{2}\left[(y - x)\mathbf{p}(x) + \int_x^y \mathbf{p}(z)\mathrm{d}z\right] \\ w_2(x, y) &= \frac{\partial w(x, y)}{\partial y} \\ &= \frac{1}{2}\left[(y - x)\mathbf{p}(y) + \int_x^y \mathbf{p}(z)\mathrm{d}z\right] \end{aligned}$$

in order to find local minima of the continuous AED-single problem.

Lemma 4: Suppose that $F_1, \ldots F_{K-1}$ is a solution of continuous AED-single, that is, an assignment that minimizes cAED. Then for every $i \leq K - 1$, $w_2(F_{i-1}, F_i) + w_1(F_i, F_{i+1}) = 0$.

Proof: Fix $i \leq K-1$, and notice that the expression above is the derivative of $w(F_{i-1}, F_i) + w(F_i, F_{i+1})$ with respect to F_i. If $w_2(F_{i-1}, F_i) + w_1(F_i, F_{i+1}) < 0$, we have that

$$w(F_{i-1}, F_i+\epsilon)+w(F_i+\epsilon, F_{i+1}) < w(F_{i-1}, F_i)+w(F_i, F_{i+1})$$

for small enough $\epsilon < 0$. If $w_2(F_{i-1}, F_i) + w_1(F_i, F_{i+1}) > 0$, the same holds for $F_i - \epsilon$. ∎

Lemma 5: Given $x < y$ in $[0, N]$, the unique z in $[y, N]$ such that $w_2(x, y) + w_1(y, z) = 0$ can be found by binary search if it exists.

Proof: Notice that $w_2(x, y) > 0$, $w_1(y, y) = 0$ and $w_1(y, z)$ *strictly decreases continuously* as z increases. Thus if $w_2(x, y) + w_1(y, N) > 0$ there is no such z, while if $w_2(x, y) + w_1(y, N) < 0$, such a z exists by the intermediate zero theorem. Since $w_2(x, y) + w_1(y, z)$ is (strictly) monotonous such a z is unique and can be found by binary search in the interval $[y, N]$. ∎

Fix the first border F_1 of a solution of continuous AED-single. Then the previous lemma can be applied repeatedly in order to determine *uniquely* the subsequent borders F_i for $i = 2 \dots K - 1$ with any arbitrary precision ϵ. Furthermore, the error can be bounded by means of the following lemma.

Algorithm 2: CONT-AED

Input: $\mathbf{p}$, N, K, ϵ
Output: F_i, $i = 0 \dots K$

$F_0 = 0$;
$a_1 = 0$;
$b_1 = N$;
while $b_1 - a_1 > \frac{\epsilon}{2K}$ **do**
  $F_1 = \frac{a_1 + b_1}{2}$;
  for $2 \le k \le K$ **do**
    $a_k = F_{k-1}$;
    $b_k = N$;
    while $b_k - a_k > \frac{\epsilon}{2K^2}$ **do**
      $F_k = \frac{a_k + b_k}{2}$;
      if $w_2(F_{k-2}, F_{k-1}) + w_1(F_{k-1}, F_k) > 0$ **then**
        $a_k = F_k$;
      else
        $b_k = F_k$;
  if $F_K < N$ **then**
    $a_1 = F_1$;
  else
    $b_1 = F_1$;

Lemma 6: Suppose that $\mathbf{p}$ is (weakly) increasing. Given F_1, $G_1 = F_1 + \epsilon$ in $[0, N]$ let F_i, G_i for $i \le K$ be the subsequent real borders as calculated in Lemma 5. Then $G_i \le F_i + i\epsilon$ for all $i \le K$.

Proof: The worst case scenario happens when $\mathbf{p}$ is constant, since the more $\mathbf{p}$ is increasing the less the changes in F_1 (which regards smaller values) affects subsequent F_i's. If $\mathbf{p}$ is constant, $F_i = iF_1$ and $G_i = iG_1 = i(F_1 + \epsilon) = F_i + i\epsilon$. ∎

Theorem 3: Suppose that $\mathbf{p}$ is (weakly) increasing. Then an ϵ-solution for continuous AED-single can be found in $O(K \log^2(N/\epsilon)))$ time by algorithm *CONT-AED*.

Proof: We now prove that the algorithm indeed returns an ϵ-solution. Let F_i, a_i, b_i, $i \le K$ be the borders calculated by the algorithm, G_i^1, $i \le K$ be the optimal borders. Since a_1 induces a sequences of borders that terminates before N, and b_1 after N, by continuity we know that $G_1^1 \in [a_1, b_1]$ thus $\left|F_1 - G_1^1\right| \le \frac{\epsilon}{2K}$. Let now G_i^2 for $i = 2 \dots K$ be the subsequent real borders calculated from $F_0 = 0$, F_1. By Lemma 6, $\left|G_i^2 - G_i^1\right| \le \frac{i\epsilon}{2K} \le \frac{\epsilon}{2}$ for all $i \le K$.

Proceed by induction on $k \le 2$. For all $k \le K$ define G_i^k, $i = k \dots K$ as the subsequent real borders calculated from F_{k-2}, F_{k-1}. By the cycle invariant of the binary search, we know that $G_k^k \in [a_k, b_k]$ hence $\left|F_k - G_k^k\right| \le \frac{\epsilon}{2K^2}$. Again by Lemma 6, we conclude that $\left|G_i^{k+1} - G_i^k\right| \le \frac{(i-k+1)\epsilon}{2K^2} \le \frac{\epsilon}{2K}$ for all $i = k \dots K$.

Notice that for all i, $F_i = G_i^{i+1}$. Furthermore,

$$\begin{aligned}\left|G_i^{i+1} - G_i^1\right| &\le \left|G_i^2 - G_i^1\right| + \sum_{k=2}^{i} \left|G_i^{k+1} - G_i^k\right| \\ &\le \frac{\epsilon}{2} + (i-1)\frac{\epsilon}{2K} \le \frac{\epsilon}{2} + \frac{\epsilon}{2} = \epsilon\end{aligned}$$

hence F_i, $i \le K$ indeed forms an ϵ-solution.

As regard to the time complexity, the algorithm CONT-AED binary searches in $\log(NK/\epsilon)$ steps for a suitable F_1, and in every one of this steps performs K binary searches which take $\log(NK^2/\epsilon)$ steps each, for total $O(K \log^2(NK^2/\epsilon)) = O(K \log^2(N/\epsilon))$ time. ∎

A. Heuristics for the AED-Single problem and experimental tests

In order to convert the ϵ-solution found by CONT-AED into a solution for the discrete AED-single problem, we considered two strategies.

First, we can simply round every F_i, $i = 1 \dots K - 1$ to the closest integer: we call this heuristic `cont-round`.

Second, we can use the borders F_i as a hint in order to run a restricted SMAWK only on the "feasible" indexes. Given a radius δ, we say that an index i is δ-feasible with respect to the continuous solution if and only if there exists a $j = 0, \dots, K-1$ such that $|F_j - i| < \delta$. Since there are at most δK δ-feasible indexes, we can run a restricted SMAWK on the $\log^2(N)$-feasible positions in additional negligible $O(K \log^2(N))$ time, and find the optimal solution for AED-single *such that all the borders are* $\log^2(N)$*-feasible*. We call this heuristic `cont-dp`.

We compare these two heuristics with the optimal algorithms SMAWK-AED and minimum K-link path (in short, `smawk` and `k-link`) introduced in Sections II-A, II-B.

The implementation was written in Python2 and run on PyPy version 2.6.1, and is available at `www.dmi.unipg.it/ pinotti` for download. Experiments were done on a system with 8 gigabytes of RAM and a Intel Core i7 processor running at 2.9 GHz (we used only one core). The system was running OS X 10.10.5 with Darwin kernel 14.5.0. The data set p_i, $i = 1 \dots N$ was chosen according to a Zipf distribution [5], [13] with skewness θ in $[0 \dots 4]$, sorted in ascending order. Namely, $p_i = \frac{(1/i)^\theta}{\sum_{i=1}^{N}(1/i)^\theta}$ with $1 \le i \le N$. The error ϵ was set to 0.1.

In Figure 1 we tested the precision of `cont-round` against several different input parameters. We do not plot

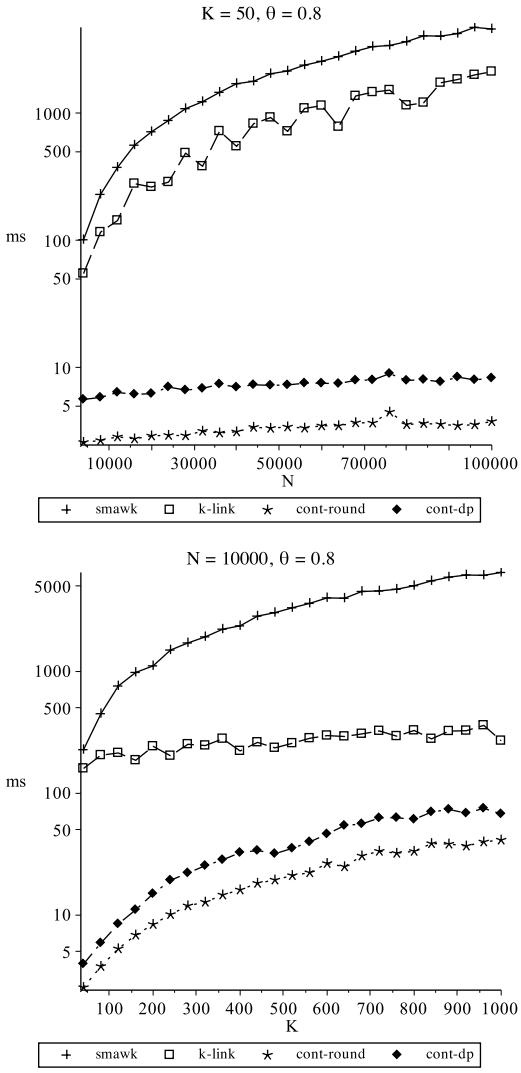

Fig. 2: Execution times of the algorithms proposed, varying N (top) and K (bottom).

the precision of strategy `cont-dp` because it was able to solve exactly all input cases we tested. The relative error of `cont-round` happened to depend linearly on K, with a bell-shaped correlation with the skewness θ and no dependence on N. All these correlations are confirmed by the border errors. The AED error was constantly lower than 0.5% on all test cases. The impact of random noise can be clearly seen due to the low total error: this happens in particular on the left side of Figure 1 where the relative error is lowest (below 10^{-7}).

In Figure 2 we compared the performance of the different exact and approximated algorithms proposed, in logarithmic scale. The algorithms were consistently sorted (from slowest to fastest) as `smawk`, `k-link`, `cont-dp` and `cont-round`, in every data set considered. As expected, `smawk` showed its linear dependence both from N and from K, while `k-link` showed its almost linear dependence on N and independence from K. Conversely, the two approximated strategies were roughly independent from N and linear on K, with `cont-dp` being twice as slow than the other. Furthermore, `cont-round` proved to be up to 1200 times faster than `smawk`, while `k-link` was significantly faster than `smawk` only for large values of K, being 24 times faster for $K = 1000$.

IV. Online algorithm

The *column-online* AED problem is so defined: Assume that the matrix $\mathbb{M}$ with K rows and N columns has already been calculated, and a new item d_{N+1} is given. Calculate the values $M[k, N+1] = opt(k, N+1)$, with $1 \le k \le K$, or equivalently, add column $(N+1)$ to matrix $\mathbb{M}$. We assume that we are searching for the optimal segmentation of the items. This version of the problem is quite useful in a dynamic context as a smart city. For example, the insertion of a new item may model a new urgent alarm or a new commercial offer that has to be broadcast on the highway or in the mall.

We now prove that column-online problem can be solved by applying the SMAWK algorithm to a suitable decomposition of $\mathbb{S}$.

Namely, fix n in $1 \dots N$ and extract from $\mathbb{S}$ the two dimensional matrix $\mathbb{S}_n = [S^k_{\ell,n}]$ for $1 \le k \le K, 1 \le \ell \le N-1$. Thus $\mathbb{S} = \cup_{n=1}^{N} \mathbb{S}_n$, that is, $\mathbb{S}$ can be decomposed into N two dimensional matrices, one for each value of n.

Adding the rightmost item d_{N+1} to previously existing items $d_1, d_2, \dots, d_N$ and computing the values $opt(k, N+1)$ with $1 \le k \le K$ using Eq. 4 may require up to $O(NK)$ comparisons. In fact for each k in $1 \dots K$, $M[k, N+1] = opt(k, N+1) = \min_{k-1 \le \ell \le N} S^k_{\ell,N+1}$ that is, $opt(k, N+1)$ corresponds to the minimum in row k of $\mathbb{S}_{N+1}$.

We now prove that the above online problem can be solved in $O(N)$ time because $\mathbb{S}_{N+1}$ is totally monotone. We first state that the matrix $\mathbb{M}$ of the classical dynamic programming, i.e., $M[k, n] = opt(k, n)$, satisfies the concave Monge property.

Lemma 7: Matrix $\mathbb{M}$ is a concave Monge matrix. ■

Corollary 1: Given $1 \le n \le N+1$, matrix $\mathbb{S}_n$ satisfies the concave Monge property and therefore it is totally monotone.

Proof: $\mathbb{S}_n$ is a concave Monge matrix if and only if $\mathbb{M}$ is a concave Monge matrix. Namely:

$$\underbrace{opt(k-1,\ell) + w(\ell, n)}_{S^k_{\ell,n}} + \underbrace{opt(k, \ell+1) + w(\ell+1, n)}_{S^{k+1}_{\ell+1,n}}$$
$$\underbrace{\le\ opt(k,\ell) + w(\ell, n)}_{S^{k+1}_{\ell,n}} + \underbrace{opt(k-1, \ell+1) + w(\ell+1, n)}_{S^k_{\ell+1,n}}$$

or equivalently, if and only if $opt(k-1, \ell) + opt(k, \ell+1) \le opt(k, \ell) + opt(k-1, \ell+1)$. ■

In conclusion:

Theorem 4: Adding a new item $N+1$ to the set of data items, the column-online AED problem can be solved in $O(N)$ time by applying the SMAWK algorithm to the two-dimensional matrix $\mathbb{S}_{N+1}$.

It is worth to point out that all the results hold both for increasing and decreasing demand probabilities since Lemma 1

does not depend on the order of the data items [1]. This can be pushed even further since matrices $\mathbb{W}$, $\mathbb{M}$, $\mathbb{S}^k$, and $\mathbb{S}_n$ are concave Monge matrices whatever is the order of the data items. Therefore, the online algorithm computes the best K-partition for any given order of the N data items, this corresponds to an optimal solution for the AED problem only if the items are sorted. If the increasing or decreasing order of the items is not preserved, the online version can be considered a fast heuristic for the allocation problem in the dynamic context. Indeed, notice that the K-link path algorithm for AED-single cannot be easily adjusted in order to accommodate the introduction of new data items, thus it would give suboptimal $O(N \log U)$ time because the solution has to be recomputed from scratch.

Also the approximated CONT-AED algorithm and the associated heuristics described in Section III cannot be adjusted. However, CONT-AED and the two heuristics can be applied from scratch and their time complexity $O(K \log^2 N/\epsilon)$ can still be competitive with the SMAWK online algorithm whenever K is $O(N^{1-\delta})$ for some $\delta > 0$. Moreover the heuristics, due to their low time complexity, can competitively recompute the solution from scratch when data items are added in arbitrary order. Recall, however, that they do not necessarily return the optimal solution for the (discrete) AED problem.

V. Conclusions

In this paper we study the AED-single and AED-full variants of the K Uniform Allocation problem with N data items and K channels, strictly improving all the solutions previously known in literature.

This problem arises in the context of quickly distributing data to multiple users via wireless communications.

The new fast algorithms were possible due to a careful analysis of the mathematical properties of the *cost functions* at hand, namely proving that the *concave Monge* property holds not only for the costs $\mathbb{W}$ but also for the solutions $\mathbb{M}$ and all the matrices involved in its computation $\mathbb{S}^k$, $\mathbb{S}_n$.

We also presented a faster $O(K \log^2(N/\epsilon))$ solution to the continuous version of the Uniform Allocation problem, which can be converted in two heuristics for the corresponding (discrete) allocation problem. Experimental results show that the cont-dp heuristic is in fact *exact* in every reasonable test data considered but slightly slower than the cont-round heuristic whose relative error depends on K and on the skewness of the data demand probabilities. An online version of the problem is also considered that allows to extend the set of items to be broadcast without recomputing the solution from scratch and guaranteeing the optimality of the solution.

References

[1] A. Aggarwal, M. Klawe, S. Moran, P. Shor, and R. Wilber. Geometric applications of a matrix-searching algorithm. *Algorithmica*, 2(1-4):195–208, 1987.

[2] A. Aggarwal, Baruch Schieber, and Takeshi Tokuyama. Finding a minimum-weight k-link path in graphs with the concave monge property and applications. *Discrete & Computational Geometry*, 12:263–280, 1994.

[1] Note that it is high probable that at the time of insertion the alarm's pressure will be maximum, while the offer's popularity will be minimum.

[3] Alok Aggarwal, Baruch Schieber, and Takashi Tokuyama. Finding a minimum weight k-link path in graphs with monge property and applications. In *Proceedings of the Ninth Annual Symposium on Computational Geometry*, SCG '93, pages 189–197, New York, NY, USA, 1993. ACM.

[4] Wolfgang Bein, Mordecai J. Golin, Lawrence L. Larmore, and Yan Zhang. The knuth-yao quadrangle-inequality speedup is a consequence of total monotonicity. *ACM Trans. Algorithms*, 6(1):17:1–17:22, December 2009.

[5] Lee Breslau, Pei Cao, Li Fan, Graham Phillips, and Scott Shenker. Web caching and zipf-like distributions: Evidence and implications. In *INFOCOM'99. Eighteenth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*, volume 1, pages 126–134, 1999.

[6] Rainer E. Burkard, Bettina Klinz, and Rüdiger Rudolf. Perspectives of Monge properties in optimization. *Discrete Applied Mathematics*, 70(2):95–161, 1996.

[7] Ardizzoni E, A. A. Bertossi, C.M. Pinotti, S. R., R. Rizzi, and M. V. S. Shashanka. Optimal Skewed Data Allocation on Multiple Channels with Flat Broadcast per Channel. *IEEE Trans. Computers*, 54(5):558–572, 2005.

[8] Paolo Ferragina, Jouni Sirén, and Rossano Venturini. Distribution-aware compressed full-text indexes. *Algorithmica*, 67(4):529–546, 2013.

[9] Zaixin Lu, Yan Shi, Weili Wu, and Bin Fu. Efficient data retrieval scheduling for multi-channel wireless data broadcast. In *INFOCOM, 2012 Proceedings IEEE*, pages 891–899, 2012.

[10] Baruch Schieber. Computing a minimum weight k-link path in graphs with the concave monge property. *J. Algorithms*, 29(2):204–222, 1998.

[11] Wai Gen Yee, Shamkant B Navathe, Edward Omiecinski, and Chris Jermaine. Efficient data allocation over multiple channels at broadcast servers. *Computers, IEEE Transactions on*, 51(10):1231–1236, 2002.

[12] Song-Yi Yi, Seunghoon Nam, and Sungwon Jung. Effective generation of data broadcast schedules with different allocation numbers for multiple wireless channels. *Knowledge and Data Engineering, IEEE Transactions on*, 20(5):668–677, 2008.

[13] George Kingsley Zipf. Human behavior and the principle of least effort. 1949.

Appendix: Algorithm SMAWK

Let A be an $m \times n$, with $m \leq n$, matrix and assume that we can compute or access each $A[i, j]$ in constant time, for any i, j. Let A be totally monotone, that is, by Def. 2, for all $a < b$ and $c < d$, $A[a, d] \leq A[a, c]$ implies $A[b, d] \leq A[b, c]$.

We describe how to compute the row-minima of A by the SMAWK algorithm.

The main block of SMAWK is the subroutine *Reduce*. It takes as input matrix A and returns an $m \times m$ submatrix $G \subset A$ that contains the columns of A which have the row-minima of A. Let k be a column index of G. Initially $k = 1$. Reduce maintains the invariant on k that $G[1 : j-1, j]$ cannot contain row-minima, for all $1 \leq j \leq k$. Also, only columns that do not contain row-minima are deleted. If $G[k, k] < G[k, k+1]$ then $G[1 : k, k+1]$ cannot contain row-minima by total monotonicity. Therefore, if $k < m$, k is increased by 1. If $k = m$, column $k+1$ cannot contain row-minima and it is deleted. Thus, k is unchanged. If $G[k, k] \geq G[k, k+1]$ then $G[k : m, k]$ cannot contain row-minima by total monotonicity. Since $G[1 : k-1, k]$ was already excluded by the invariant, column k is deleted because it cannot contain the row-minima. Then, k is set to the minimum between 1 and $k-1$.

The row-minima of an $m \times n$ totally monotone matrix A with $m \leq n$ are then found as follows. Invoke Reduce on A to get an $m \times m$ matrix G, and then recursively find the row-minima of the submatrix of G which consists of the even rows of G. After having found the row-minima of even rows of G, compute the row-minima in odd rows.