

Unbiased Rounding of Rational Matrices

Benjamin Doerr and Christian Klein

Max-Planck-Institut für Informatik, Saarbrücken, Germany

Abstract. Rounding a real-valued matrix to an integer one such that the rounding errors in all rows and columns are less than one is a classical problem. It has been applied to hypergraph coloring, in scheduling and in statistics. Here, it often is also desirable to round each entry randomly such that the probability of rounding it up equals its fractional part. This is known as unbiased rounding in statistics and as randomized rounding in computer science.

We show how to compute such an unbiased rounding of an $m \times n$ matrix in expected time $O(mnq^2)$, where q is the common denominator of the matrix entries. We also show that if the denominator can be written as $q = \prod_{i=1}^{\ell} q_i$ for some integers q_i , the expected runtime can be reduced to $O(mn \sum_{i=1}^{\ell} q_i^2)$. Our algorithm can be derandomised efficiently using the method of conditional probabilities.

Our roundings have the additional property that the errors in all initial intervals of rows and columns are less than one.

1 Introduction

In this paper, we analyze a rounding problem with strong connections to statistics, but also to different areas in discrete mathematics, computer science, and operations research. We present an efficient way to round a matrix to an integer one such that the rounding errors in all intervals (i.e., a set of consecutive entries) of rows and columns are small.

For real numbers a, b let $[a..b] := \{z \in \mathbb{Z} \mid a \leq z \leq b\}$. For $x \in \mathbb{R}$ let $\lfloor x \rfloor := \max\{z \in \mathbb{Z} \mid z \leq x\}$, $\lceil x \rceil := \min\{z \in \mathbb{Z} \mid z \geq x\}$ and $\{x\} := x - \lfloor x \rfloor$. For $q \in \mathbb{N}$ let $\frac{1}{q}\mathbb{Z} := \{\frac{p}{q} \mid p \in \mathbb{Z}\}$. We show the following.

Theorem 1. For all $X \in \frac{1}{q}\mathbb{Z}^{m \times n}$ a randomized rounding $Y \in \mathbb{Z}^{m \times n}$ such that

$$\forall b \in [1..n], i \in [1..m] : \left| \sum_{j=1}^b (x_{ij} - y_{ij}) \right| < 1, \quad (1)$$

$$\forall b \in [1..m], j \in [1..n] : \left| \sum_{i=1}^b (x_{ij} - y_{ij}) \right| < 1, \quad (2)$$

$$\left| \sum_{i=1}^m \sum_{j=1}^n (x_{ij} - y_{ij}) \right| < 1 \quad (3)$$

can be computed in expected time $O(mn \sum_{i=1}^{\ell} p_i^2)$, where $q = \prod_{i=1}^{\ell} p_i$, $p_i \in \mathbb{N}$ is a factorization of q .

The result above can be derandomised using the method of conditional probabilities, leading to a deterministic algorithm having asymptotically the same runtime.

Theorem 2. For all $X \in \frac{1}{q}\mathbb{Z}^{m \times n}$ a rounding $Y \in \mathbb{Z}^{m \times n}$ such that the inequalities (1), (2) and (3) in Theorem 1 hold, can be computed in time $O(mn \sum_{i=1}^{\ell} p_i^2)$, where $q = \prod_{i=1}^{\ell} p_i$, $p_i \in \mathbb{N}$ is a factorization of q .

Previous results on this particular rounding problem were given by Doerr et al. in [7]. Theorem 2 extends their result to arbitrary rational matrices. Their equivalent of Theorem 1, however, only works for matrices of numbers with finite binary expansion. Hence, they cannot, for example, round decimal fractions, as is often required in applications. For deterministic rounding they give an $O(mn \log(mn))$ time algorithm, while our algorithm is linear in the matrix size.

1.1 Baranyai's Rounding Lemma and Applications in Statistics

Baranyai [2] used a weaker variant of Theorem 2 to obtain his famous results on coloring and partitioning complete uniform hypergraphs. He showed that any matrix can be rounded such that the errors in all rows, all columns and the whole matrix are less than one. He used a formulation as flow problem to prove this statement, giving super-linear runtime. However, algorithmic issues were not his focus.

In statistics, Baranyai's result was independently obtained by Bacharach [1] (in a slightly weaker form), by Causey, Cox and Ernst [3], and, again independently, by Šíma [10]. There are two statistics applications for such rounding results. Note first that instead of rounding fractions to integers, our result also applies to rounding to multiples of any other integer (e.g., multiples of 10). Such a rounding can be used to improve the readability of data tables.

The main reason, however, to apply such a rounding procedure is confidentiality protection. Frequency counts that directly or indirectly disclose small counts may permit the identification of individual respondents. There are various methods to prevent this [12], one of which is *controlled rounding* [5]. Here, one tries to round an $(m+1) \times (n+1)$ -table \tilde{X} given by

$$\begin{array}{c|c} (x_{ij})_{\substack{i=1\dots m \\ j=1\dots n}} & \left(\sum_{j=1}^n x_{ij} \right)_{i=1\dots m} \\ \hline \left(\sum_{i=1}^m x_{ij} \right)_{j=1\dots n} & \sum_{i=1}^m \sum_{j=1}^n x_{ij} \end{array}$$

to an $(m+1) \times (n+1)$ -table \tilde{Y} such that additivity is preserved, i.e., the last row and column of \tilde{Y} contain the associated totals of \tilde{Y} . In our setting we round the $m \times n$ -matrix X defined by the mn inner cells of the table \tilde{X} to obtain a controlled rounding.

The additivity in the rounded table allows to derive information on the row and column totals of the original table. In contrast to previous rounding algorithms, our result also permits to retrieve further reliable information from the rounded matrix, namely on the sums of consecutive elements in rows or columns. Such queries make sense if there is a linear ordering on statistical attributes.

Here is an example. Let x_{ij} be the number of people in country i that are j years old. Say Y is such that $\frac{1}{1000}Y$ is a rounding of $\frac{1}{1000}X$ as in Theorem 1. Now $\sum_{j=20}^{40} y_{ij}$ is the number of people in country i that are between 20 and 40 years old, apart from an error of less than 2000. Note that such guarantees are not provided by Baranyai [2], Bacharach [1], and Causey, Cox and Ernst [3].

1.2 Unbiased and Randomized Rounding

Our randomized algorithm has the additional property that each matrix entry is rounded up with probability equal to its fractional value. This is known as *randomized rounding* in computer science [9] and as *unbiased rounding* in statistics [4, 8]. Here, a controlled rounding is computed such that the expected values of each table entry (including the totals) equals its fractional value in the original table.

Definition 1. Let $x \in \mathbb{R}$. A random variable y is called randomized rounding of x , denoted by $y \approx x$, if $\Pr(y = \lfloor x \rfloor + 1) = \{x\}$ and $\Pr(y = \lfloor x \rfloor) = 1 - \{x\}$. For a matrix $X \in \mathbb{R}^{m \times n}$, we call a $\mathbb{Z}^{m \times n}$ -valued random variable Y randomized rounding of X if $y_{ij} \approx x_{ij}$ for all $i \in [1..m], j \in [1..n]$.

Note that if $y \approx x$, then $\Pr(|y - x| < 1) = 1$ and $\mathbb{E}(y) = x$. In fact, the converse holds as well. Hence we can restate Theorem 1 in the following stronger form.

Theorem 3. For all $X \in \frac{1}{q}\mathbb{Z}^{m \times n}$ a randomized rounding $Y \in \mathbb{Z}^{m \times n}$ fulfilling the additional constraints

$$\begin{aligned} \forall b \in [1..n], i \in [1..m] : \sum_{j=1}^b x_{ij} &\approx \sum_{j=1}^b y_{ij}, \\ \forall b \in [1..m], j \in [1..n] : \sum_{i=1}^b x_{ij} &\approx \sum_{i=1}^b y_{ij}, \\ \sum_{i=1}^m \sum_{j=1}^n x_{ij} &\approx \sum_{i=1}^m \sum_{j=1}^n y_{ij} \end{aligned}$$

can be computed in expected time $O(mn \sum_{i=1}^{\ell} p_i^2)$, where $q = \prod_{i=1}^{\ell} p_i$, $p_i \in \mathbb{N}$ is a factorization of q .

2 Preliminaries

2.1 Random Walks

We need some well known facts about one-dimensional random walks with absorbing barriers. Consider a set of $n + 1$ vertices labeled v_0 to v_n . From vertex v_i , $i \in [1..n - 1]$, one can either take a step to vertex v_{i+1} or v_{i-1} , both with probability $\frac{1}{2}$. At the endpoints v_0 and v_n , no further steps can be taken. We write $\Pr(v_i \nearrow v_n)$ for the probability that a random walk from vertex v_i will reach v_n instead of v_0 and $\mathbb{E}(\text{Steps}(v_i))$ for the expected number of steps a random walk starting in vertex v_i needs to reach either vertex v_0 or v_n .

Lemma 1. $\Pr(v_i \nearrow v_n) = \frac{i}{n}$.

Proof. From the definition of random walks we obtain the equations $\Pr(v_n \nearrow v_n) = 1$, $\Pr(v_i \nearrow v_n) = \frac{1}{2}\Pr(v_{i-1} \nearrow v_n) + \frac{1}{2}\Pr(v_{i+1} \nearrow v_n)$, $i \in [1..n-1]$, and $\Pr(v_0 \nearrow v_n) = 0$. It can easily be checked that this system of equations has the unique solution $\Pr(v_i \nearrow v_n) = \frac{i}{n}$. \square

Lemma 2. $\mathbb{E}(\text{Steps}(v_i)) = i(n-i)$.

Proof. Again, we obtain the system of equations $\mathbb{E}(\text{Steps}(v_0)) = \mathbb{E}(\text{Steps}(v_n)) = 0$ and $\mathbb{E}(\text{Steps}(v_i)) = 1 + \frac{1}{2}\mathbb{E}(\text{Steps}(v_{i-1})) + \frac{1}{2}\mathbb{E}(\text{Steps}(v_{i+1}))$, $i \in [1..n-1]$. It can easily be checked that this system of equations has the unique solution $\mathbb{E}(\text{Steps}(v_i)) = i(n-i)$. \square

2.2 Integrality of Row and Column Sums

In the following we always assume the input matrix X to be from $[0, 1]^{m \times n}$. Otherwise, simply subtract the integral part of X before rounding and add it again afterwards. Furthermore, we assume X to have integral row and column sums, as justified by the following lemma from [7].

Lemma 3. *Assume that for any $X \in \mathbb{R}^{m \times n}$ with integral row and column sums, a rounding $Y \in \mathbb{Z}^{m \times n}$ satisfying inequality (1) and (2) from Theorem 1 can be computed in time $T(m, n)$. Then for all $\tilde{X} \in \mathbb{R}^{m \times n}$ with arbitrary row and column sums, a rounding $\tilde{Y} \in \mathbb{Z}^{m \times n}$ satisfying inequalities (1), (2) and (3) can be computed in time $T(m+1, n+1) + O(mn)$.*

3 Unbiased Rounding

3.1 Index Intervals

What properties does a rounding Y of X fulfilling the inequalities of Theorem 1 have? Substituting $b = n$ in inequality (1), we can deduce that the i th row of Y must contain exactly $\sum_{j=1}^n x_{ij}$ many 1-entries. To fulfill the inequality for $b \neq n$, there must be $\lfloor \sum_{j=1}^b x_{ij} \rfloor$ or $\lceil \sum_{j=1}^b x_{ij} \rceil$ many 1-entries in column 1 to b of the i th row of Y . Inequality (2) gives analogous statements for columns. This observation suggests that we should put one 1 in each interval bounded by two positions where the integral part of the partial row (resp. column) sum increases. This motivates the following definition.

We define the k th index interval of the i th row of X as

$$I_i^X(k) := \left\{ j \in [1..n] \mid x_{ij} \neq 0 \wedge \sum_{\ell=1}^j x_{i\ell} > k-1 \wedge \sum_{\ell=1}^{j-1} x_{i\ell} < k \right\}.$$

Column index intervals are defined analogous. Observe that the sum over all entries of an index interval is at least one. Because of this, each index interval consists of at least two non-zero elements. If the sum is more than one, the interval shares an entry

with an neighboring interval. The following example shows a row of values and the corresponding index intervals.

$$\overbrace{0.2 \quad 0.7}^{I(1)} \quad \overbrace{0.8 \quad 0.6}^{I(2)} \quad \overbrace{0.4 \quad 0.3}^{I(3)} \quad \overbrace{0.5 \quad 0.4 \quad 0.1}^{I(4)}$$

The idea now is to “concentrate the total value of all entries” of an index interval into a single entry until it has value 1. For this, observe that if we pick two non-zero entries in the same row index interval and modify one by $+\frac{1}{q}$ and the other by $-\frac{1}{q}$, we don’t change any of the partial sums left of the first or right of the second entry. In particular, the total sum of this row stays unchanged. The same holds for columns.

3.2 The Algorithm

```

ROUND( $X \in (\frac{1}{q}\mathbb{Z} \cap [0, 1])^{m \times n}$ )
1   $t \leftarrow 0$ 
2   $X^{(0)} \leftarrow X$ 
3  Compute row and column index intervals of  $X^{(0)}$ 
4  while  $X^{(t)} \notin \{0, 1\}^{m \times n}$ 
5      do
6           $C \leftarrow \text{FINDCYCLE}(X^{(t)})$ 
7          Choose  $a \in \{+\frac{1}{q}, -\frac{1}{q}\}$ 
8           $X^{(t+1)} \leftarrow$  alternately augment  $X^{(t)}$  along  $C$  by  $\pm a$ 
9           $t \leftarrow t + 1$ 
10         Update row and column index intervals of  $X^{(t)}$ .
11  return  $Y := X^{(t)}$ 

```

Fig. 1. The rounding algorithm.

The algorithm now iteratively modifies the matrix until all elements are 0 or 1. In each step it first constructs a cycle in the current matrix that alternately pairs two directly adjacent fractional elements in the same row interval resp. column interval¹. This way each element of the cycle has one horizontal and one vertical neighbor in the cycle. How to construct such cycles will be discussed in Section 3.4. The algorithm then traverses this cycle and alternately adds $\frac{1}{q}$ and subtracts $\frac{1}{q}$ to each cycle entry.

The current matrix is stored in a two-dimensional doubly linked list where every non-integral entry has a pointer to the next non-integral entry in each direction. For the cycle finding step the algorithm must keep track of the index intervals of the current matrix $X^{(t)}$. To do this, the fractional parts $s_{ij}^{col} := \{\sum_{k=1}^i x_{kj}\}$ and $s_{ij}^{row} := \{\sum_{k=1}^j x_{ik}\}$ of

¹ There is a special case where this is not true, as we will see later.

the partial row and column sums of each entry $x_{ij}, i \in [1..m], j \in [1..n]$ are computed for the initial matrix and updated during the augmentation step. With these values the algorithm can decide if the neighbor of an entry belongs to the same index interval or not, based on the value of the neighbor entry and on the fractional part of the current entry. Whenever two neighboring elements inside the same index interval are augmented, only their partial sums change, hence the cost of an update is linear in the size of the current cycle.

If an augmentation changes an element to 0 or 1, it is removed from the data structure. Also, when updating the intervals, such element are ignored. By disregarding entries changed to 1, the corresponding row and column sums decrease by 1 and thus also the number of intervals decreases by 1 if this happens. Since the fractional part of an element that changes to 0 or 1 is also 0, this does not change the values s_{ij}^{col} or s_{ij}^{row} for any other element.

3.3 Runtime and Unbiasedness

For the moment let us assume that the call in Line 6 of the algorithm always returns a cycle and takes time proportional to the cycle size. Does the algorithm terminate? As we will see, this depends on how we choose a in Line 7. Each value for a corresponds to one of the two possible choices we have when doing the augmentation along the cycle. Either we start by adding $+\frac{1}{q}$ to the first element on the cycle, then $-\frac{1}{q}$ to the second and so on, or we start by adding $-\frac{1}{q}$ then $+\frac{1}{q}$ and so on. If one of this possibilities is chosen uniformly at random, we have the following theorem.

Theorem 4. *Assume that in Line 7 of the algorithm from Figure 1, a is chosen independently at random such that $\Pr(a = \frac{1}{q}) = \Pr(a = -\frac{1}{q}) = \frac{1}{2}$. Then the following holds.*

- *The algorithm terminates in expected time $O(mnq^2)$.*
- *Each $x_{ij}, i \in [1..m], j \in [1..n]$ is rounded to one with probability x_{ij} .*

Proof. Consider an element $x_{ij}, i \in [1..m], j \in [1..n]$ of the cycle. With probability $\frac{1}{2}$ each, we will either add or subtract $\frac{1}{q}$ from it. But this is equivalent to doing a random walk on a line with $q + 1$ elements, starting from position $q \cdot x_{ij}$. From Lemma 2 it follows that the element becomes 0 or 1 after an expected number of $O(q^2)$ augmentations. As soon as this happens, x_{ij} will no longer belong to any index interval, and hence will no longer be chosen during the cycle construction. Since the matrix has mn entries, the first claim follows. The second claim follows immediately from Lemma 1. \square

3.4 Finding Cycles

We now specify the function FINDCYCLE used by the algorithm to find a cycle along which it can round. As we will see, the fact that we aim at low errors in all initial intervals (and not only whole rows and columns) imposes some subtle additional difficulties.

First an arbitrary non-integral matrix entry a_1 is chosen as current entry. Then, alternately pick a non-integral entry directly adjacent to the current entry in the same

row interval resp. in the same column interval as new current entry. This way, a sequence (a_1, \dots, a_ℓ) of matrix entries is constructed. Since each index interval contains at least two fractional entries, the cycle construction routine can not fail to construct a cycle C as long as the matrix is not integral. The algorithm stops as soon as an element already picked before, say $a_k, k \in [1..l-1]$, can be chosen as current element. Assume

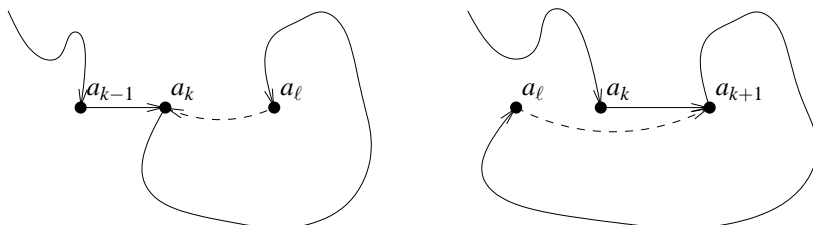


Fig. 2. The two possibilities during cycle construction.

that a_k and a_ℓ share a row interval². By construction, either a_{k-1} or a_{k+1} will also be an element of this row. If a_{k-1} is an element of this row, $C := (a_k, \dots, a_\ell)$ is a cycle alternatingly pairing row and column elements sharing common intervals as needed by the main algorithm. However, if a_{k+1} is an element of this row, the above cycle would contain two successive edges pairing row entries, namely (a_ℓ, a_k) and (a_k, a_{k+1}) .

In this case, the cycle $C := (a_{k+1}, \dots, a_\ell)$ is chosen instead which again alternatingly pairs row and column elements (See Figure 2 for details.). As this cycle now contains an edge pairing an element to its neighbors neighbor, the algorithm has to modify one additional partial sum during the augmentation, namely the one of a_k by $\pm \frac{1}{q}$ depending on how the pair (a_ℓ, a_{k+1}) is augmented. Observe that if a_k belongs to two overlapping index intervals, then a_ℓ and a_{k+1} belong to different intervals. As we will see in the analysis, this will not influence the correctness of the algorithm.

We finally argue that FINDCYCLE has an amortized runtime of $\Theta(|C|)$, where $|C|$ is the length of the cycle computed. Because augmenting along C only changes the local structure between two paired elements, the remaining elements of the sequence that were not chosen for C still alternatingly connect entries of the same row resp. column interval. Hence, the next time FINDCYCLE is called, it can reuse the part of the sequence not used to construct the cycle. Thus, over the whole algorithm, each element is touched during cycle construction as often as it is part of a cycle.

3.5 Correctness

In the following we only consider rows, as the arguments for columns are analogous. Hence, let $(x_1^{(t)}, \dots, x_n^{(t)}) := (x_{i1}^{(t)}, \dots, x_{in}^{(t)})$ be the elements of the i th row of $X^{(t)}$, for an arbitrary $i \in [1..m]$. Let $I^{(t)}(1), \dots, I^{(t)}(k)$ be the $k := \sum_{j=1}^n x_j^{(t)}$ index intervals of this row. For $\ell \in [1..k]$, we write $L(I^{(t)}(\ell)) := \min(I^{(t)}(\ell))$ and $R(I^{(t)}(\ell)) := \max(I^{(t)}(\ell))$ for

² Again the same holds for columns.

the position of the leftmost, resp. rightmost entry of the ℓ th interval. If $L(I^{(t)})(\ell)$ (resp. $R(I^{(t)})(\ell)$) does not belong to two intervals, we call it *proper*. If both $L(I^{(t)})(\ell)$ and $R(I^{(t)})(\ell)$ are proper, we call $I^{(t)}(\ell)$ *proper*.

The interior $I^{(t)}(\ell)^\circ$ of an interval is defined as the set of all elements that only belong to this interval. Hence, $I^{(t)}(\ell)^\circ = I^{(t)}(\ell)$ if and only if the interval is proper.

By the definition of index intervals, $R(I^{(t)})(\ell)$ is proper if and only if the partial sum up to this entry is integral. $L(I^{(t)})(\ell)$ is proper if and only if $R(I^{(t)})(\ell - 1)$ is proper. Hence, $I^{(t)}(\ell)$ is proper if the sum over all entries in $I^{(t)}(\ell)$ is 1.

In the special case where we constructed a cycle pairing two entries $x_a^{(t)}, x_b^{(t)}$ from neighboring row intervals, those intervals share a common element $x_j^{(t)} \neq 0, j \in [a..b]$. Augmenting along this cycle introduces no inconsistencies in the columns, as all other pairs of entries are taken from a common interval. Modifying $x_a^{(t)}, x_b^{(t)}$ to, say, $x_a^{(t)} + \frac{1}{q}, x_b^{(t)} - \frac{1}{q}$, can, for the analysis, be viewed as modifying $x_a^{(t)} + \frac{1}{q}, x_j^{(t)} - \frac{1}{q}$ and $x_j^{(t)} + \frac{1}{q}, x_b^{(t)} - \frac{1}{q}$ independently. Since $x_j^{(t)}$ is a non-zero multiple of $\frac{1}{q}$ shared by both intervals, this is always possible.

First we show that as long as no element of an interval is set to one, the interval will only contract.

Lemma 4. *Let $I^{(t)}(\ell)$ be the ℓ th interval at time t . Assume that no entry of $I^{(t)}(\ell)$ changes to 1. Let $I^{(t+1)}(\ell')$ be an interval at time $t + 1$ that intersects $I^{(t)}(\ell)$.*

- a) *If $R(I^{(t)})(\ell)$ is proper, then $R(I^{(t+1)})(\ell')$ is proper and $R(I^{(t)})(\ell) \geq R(I^{(t+1)})(\ell')$.*
- b) *If $L(I^{(t)})(\ell)$ is proper, then $L(I^{(t+1)})(\ell')$ is proper and $L(I^{(t)})(\ell) \leq L(I^{(t+1)})(\ell')$.*
- c) *If $L(I^{(t)})(\ell)$ and $L(I^{(t+1)})(\ell')$ are not proper, then $L(I^{(t)})(\ell) = L(I^{(t+1)})(\ell')$.*
- d) *If $L(I^{(t)})(\ell)$ is not proper, but $L(I^{(t+1)})(\ell')$ is proper, then $R(I^{(t+1)})(\ell' - 1) \leq R(I^{(t)})(\ell) = L(I^{(t)})(\ell) \leq L(I^{(t+1)})(\ell')$.*

Proof. First observe that if $L(I^{(t)})(\ell)$ (resp. $R(I^{(t)})(\ell)$) is proper, it can only be paired with an element to its right (left). Since augmenting a pair does not change the partial sum of its right entry, we get the first statement. For the second statement observe that the partial sum up to $L(I^{(t)})(\ell)$ has the same fractional value as this element. Hence before it can be made small enough to belong to the $(\ell - 1)$ th interval, it will be zero, since all changes are done in steps of $\frac{1}{q}$. Statement c) follows from the fact that a shared element always has value larger than $\frac{1}{q}$. Since the augmentation only changes each value by at most $\frac{1}{q}$, this also proves d). \square

Lemma 5. *Let $I^{(t)}(\ell)$ be the ℓ th interval at time t and let $x_a^{(t)}$ be an element of $I^{(t)}(\ell)$ that changes to 1. If $x_a^{(t)}$ is shared with $I^{(t)}(\ell + 1)$, both intervals will merge. Otherwise $I^{(t)}(\ell)$ vanishes and both the rightmost border of the interval to the left as well as the leftmost border of the interval to the right become proper.*

Proof. Surely $x_a^{(t)} = \frac{q-1}{q}$.

First, assume that $x_a^{(t)}$ is shared with $I^{(t)}(\ell + 1)$. Then the partial sum for $x_a^{(t)}$ must have fractional value smaller than $x_a^{(t)}$, hence the intervals will merge.

Now assume that $x_a^{(t)}$ is an inner element of $I^{(t)}(\ell)$. Then the partial sum for $x_a^{(t)}$ must either be $\frac{q-1}{q}$ and $L(I^{(t)})(\ell) = a$ is proper, or it must be integral and $R(I^{(t)})(\ell) = a$ is proper. Note that in both cases the other border of $I^{(t)}(\ell)$ is the only non-integral element of this interval. If this element is also proper, the lemma obviously holds, hence assume it is shared (and thus has value at least $\frac{2}{q}$). If $a = L(I^{(t)})(\ell)$, then the augmentation will cause the partial sum for $x_a^{(t)}$ to become integral, making $R(I^{(t)})(\ell)$ the proper left border of $I^{(t)}(\ell + 1)$. Otherwise the augmentation will cause the partial sum for $L(I^{(t)})(\ell)$ to become integral, making it the proper right border of $I^{(t)}(\ell - 1)$. \square

Lemma 6. *Let Y be a rounding of X computed by the algorithm in Figure 1. Then for each row there exists a bijective mapping between elements rounded to 1 and index intervals of this row in X , mapping each element to an interval containing it. The same holds for columns.*

Proof. Let $K := (I_a, \dots, I_b)$ be a maximum collection of neighboring intervals in an arbitrary row of X such that $I_j \cap I_{j+1} \neq \emptyset$ for $j \in [a..(b-1)]$. In other words, exactly $L(I_a)$ and $R(I_b)$ are proper. Clearly, it suffices to prove the lemma for such subcollections.

First assume that at time t no element is changed to 1. If none of the shared borders of intervals in K become proper, then nothing changes according to Lemma 4. Otherwise, K decomposes into smaller collections of intervals which can be treated separately.

Now assume that at time t an inner element $x_j^{(t)}$ of a current interval changes to 1. By Lemma 5, this interval was obtained by merging $d - c + 1$ neighboring intervals $I_c, \dots, I_d, a \leq c \leq d \leq b$ of the initial collection. This means that their $d - c$ shared entries were set to 1 during the algorithm. Hence we can assign 1 to the interval of the initial collection containing $x_j^{(t)}$, and the remaining $d - c$ 1s to the other intervals. Since by Lemma 5 the borders of the neighbors of this interval become proper in this case, we get two smaller subcollections which can be treated separately. \square

Theorem 5. *If Y is a rounding of X computed by the algorithm in Figure 1, then*

$$\forall b \in [1..n], i \in [1..m] : \left| \sum_{j=1}^b (x_{ij} - y_{ij}) \right| < 1,$$

$$\forall b \in [1..m], j \in [1..n] : \left| \sum_{i=1}^b (x_{ij} - y_{ij}) \right| < 1.$$

Proof. Let $b \in [1..n]$ and $i \in [1..m]$. If $x_{ib} = 0$ then $y_{ib} = 0$. Hence it suffices to regard the case $x_{ib} \neq 0$. Let $\ell \in \mathbb{N}$ be maximal such that x_{ib} is contained in the ℓ th interval of the i th row of X . By definition, this means that $\ell - 1 < \sum_{j=1}^b x_{ij} \leq \ell$. By Lemma 6, $\ell - 1 \leq \sum_{j=1}^b y_{ij} \leq \ell$ holds. If $\sum_{j=1}^b x_{ij} < \ell$, this shows the theorem. In the other case, x_{ib} must be the last element of the ℓ th interval, hence $\sum_{j=1}^b y_{ij} = \ell$. For columns, the proof is analogous. \square

4 Iterative Rounding

If q has a non-trivial factorization $q = q_1 \cdot q_2$ with $q_1, q_2 \in \mathbb{N}_{\geq 2}$, this can be exploited to improve the runtime from Theorem 4. Our approach resembles that given by Doerr [6] for powers of 2.

```

COMPUTEROUNDING( $X \in \frac{1}{q_1 \cdot q_2} \mathbb{Z}^{m \times n}$ )
1  Compute  $X' \in \frac{1}{q_1} \mathbb{Z}^{m \times n}, X'' \in \frac{1}{q_2} \mathbb{Z}^{m \times n}$  such that  $X = X' + \frac{1}{q_1} X''$ 
2   $Y'' \leftarrow \text{ROUND}(X'') \in \{0, 1\}^{m \times n}$ 
3   $\tilde{X} \leftarrow X' + \frac{1}{q_1} Y'' \in \frac{1}{q_1} \mathbb{Z}^{m \times n}$ 
4   $Y \leftarrow \text{ROUND}(\tilde{X}) \in \{0, 1\}^{m \times n}$ 
5  return  $Y \in \{0, 1\}^{m \times n}$ 

```

Fig. 3. The factor rounding algorithm.

Lemma 7. *Let $X \in \frac{1}{q} \mathbb{Z}^{m \times n}$ be a rational matrix with $q = q_1 q_2$ and $q_1, q_2 \in \mathbb{N}$. Then COMPUTEROUNDING in Figure 3 will compute an unbiased rounding Y of X satisfying*

$$\forall b \in [1..n], i \in [1..m] : \left| \sum_{j=1}^b (x_{ij} - y_{ij}) \right| < 1,$$

$$\forall b \in [1..m], j \in [1..n] : \left| \sum_{i=1}^b (x_{ij} - y_{ij}) \right| < 1.$$

Proof. First note that the algorithm decomposes each matrix entry $x_{ij}, i \in [1..m], j \in [1..n]$ into $x'_{ij} \in \frac{1}{q_1} \mathbb{Z}$ and $x''_{ij} \in \frac{1}{q_2} \mathbb{Z}$. To show unbiasedness, observe that in Line 2, an unbiased rounding $y''_{ij} \in \{0, 1\}$ of x''_{ij} is computed according to Theorem 4. In other words, \tilde{x}_{ij} computed in Line 3 will have value $x'_{ij} + \frac{1}{q_1}$ with probability x''_{ij} , and value x'_{ij} otherwise. From Line 4 it follows that

$$\begin{aligned} \Pr(y_{ij} = 1) &= \Pr(\tilde{x}_{ij} \nearrow 1) = x''_{ij} \Pr\left(x'_{ij} + \frac{1}{q_1} \nearrow 1\right) + (1 - x''_{ij}) \Pr(x'_{ij} \nearrow 1) \\ &= x''_{ij} \left(x'_{ij} + \frac{1}{q_1}\right) + (1 - x''_{ij}) x'_{ij} \\ &= \frac{1}{q_1} x''_{ij} + x'_{ij} = x_{ij}, \end{aligned}$$

hence the algorithm computes an unbiased rounding of X .

To see that the rounding computed in Figure 3 is a controlled rounding satisfying our additional constraints, let $s_{ij}(X) := \sum_{k=1}^i x_{kj}$ for $i \in [1..m], j \in [1..n]$, be the sum over the first i elements of the j th column of X . In Line 2, a controlled rounding Y'' of X'' satisfying our additional constraints is computed, hence $|s_{ij}(X'' - Y'')| \leq 1 - \frac{1}{q_2}$. A

similar statement holds for Y and \tilde{X} in Line 4, namely $|s_{ij}(\tilde{X} - Y)| \leq 1 - \frac{1}{q_1}$. These two error bounds and the triangle inequality yield

$$\begin{aligned} |s_{ij}(X - Y)| &= |s_{ij}(X' + \frac{1}{q_1}X'' - \frac{1}{q_1}Y'' + \frac{1}{q_1}Y'' - Y)| \\ &\leq |s_{ij}(\tilde{X} - Y)| + \frac{1}{q_1}|s_{ij}(X'' - Y'')| \\ &\leq 1 - \frac{1}{q_1} + \frac{1}{q_1}(1 - \frac{1}{q_2}) = 1 - \frac{1}{q}, \end{aligned}$$

hence the error in all initial column intervals is at most $1 - \frac{1}{q}$. The proof for the error in initial row intervals and in single elements is analogous. \square

Now let $q = \prod_{i=1}^{\ell} q_i$, $q_i \in \mathbb{N}$ be a factorization of the denominator of X . Then the algorithm in Figure 3 can be applied recursively to get the main result as stated in Theorem 1.

Since for $X \in \{0, \frac{1}{2}\}^{m \times n}$, an augmentation of an element by $\pm \frac{1}{2}$ will always change the element to either 0 or 1, the algorithm from Figure 1 will run in *deterministic* time $O(mn)$ for this special case. Using this observation and choosing $q = 2^\ell$, this gives the result from [7] for unbiased rounding of matrices of ℓ -bit numbers.

Corollary 1. *Let $X \in [0, 1]^{m \times n}$ be a matrix of ℓ -bit numbers. Then an unbiased controlled rounding of X satisfying equations (1), (2) and (3) from Theorem 1 can be computed in time $O(mn\ell)$.*

5 Derandomisation

The algorithm in Figure 1 can be derandomised using the method of conditional probabilities (cf. [11]). For this, observe that by Lemma 2 the expected number $\mathbb{E}(\text{Steps}(X))$ of augmentations needed to round a given matrix $X \in (\frac{1}{q}\mathbb{Z} \cap [0, 1])^{m \times n}$ is

$$\mathbb{E}(\text{Steps}(X)) = \sum_{i=1}^m \sum_{j=1}^n x_{ij}(q - x_{ij}) = O(mnq^2).$$

The derandomisation now works as follows. At the beginning of the algorithm in Figure 1, $\mathbb{E}(\text{Steps}(X))$ is computed. Each time one of the two possible ways to augment along a cycle C in Line 7 of the algorithm must be chosen, this isn't done randomly. Instead, the augmentation for which the algorithm would need the fewer number of expected steps if it would continue choosing randomly is picked. By Lemma 2 it follows that

$$\mathbb{E}(\text{Steps}(X)) = |C| + \frac{1}{2}\mathbb{E}(\text{Steps}(X - X_C)) + \frac{1}{2}\mathbb{E}(\text{Steps}(X + X_C)),$$

where X_C is the matrix for one of the two possible augmentations along C . From this formula it follows that $\mathbb{E}(\text{Steps}(X - X_C))$ and $\mathbb{E}(\text{Steps}(X + X_C))$ cannot both be larger than $\mathbb{E}(\text{Steps}(X)) - |C|$. Hence, each time the algorithm augments along a cycle C , $\mathbb{E}(\text{Steps}(X))$ decreases by at least $|C|$, since the augmentation giving the smaller expected value is picked.

Calculating $\mathbb{E}(\text{Steps}(X))$ for the input matrix needs time $O(mn)$. Deciding which augmentation to use for cycle C in step t can be done in time $O(|C|)$ while constructing the cycle. The value $\mathbb{E}(\text{Steps}(X^{(t)}))$ can be derived from $\mathbb{E}(\text{Steps}(X^{(t-1)}))$ in $O(|C|)$ time while doing the actual augmentation. This gives the following theorem.

Theorem 6. For all $X \in \frac{1}{q}\mathbb{Z}^{m \times n}$ a rounding $Y \in \mathbb{Z}^{m \times n}$ such that the inequalities (1), (2) and (3) from Theorem 1 hold can be computed in time $O(mnq^2)$.

Together with Lemma 7, this yields Theorem 2 from the introduction.

References

1. M. Bacharach. Matrix rounding problems. *Management Science (Ser. A)*, 12:732–742, 1966.
2. Zs. Baranyai. On the factorization of the complete uniform hypergraph. In *Infinite and finite sets (Colloq., Keszthely, 1973; dedicated to P. Erdős on his 60th birthday)*, Vol. I, pages 91–108. Colloq. Math. Soc. János Bolyai, Vol. 10. North-Holland, Amsterdam, 1975.
3. B. D. Causey, L. H. Cox, and L. R. Ernst. Applications of transportation theory to statistical problems. *Journal of the American Statistical Association*, 80:903–909, 1985.
4. L. H. Cox. A constructive procedure for unbiased controlled rounding. *Journal of the American Statistical Association*, 82:520–524, 1987.
5. L. H. Cox and L. R. Ernst. Controlled rounding. *Informes*, 20:423–432, 1982.
6. B. Doerr. Generating randomized roundings with cardinality constraints and derandomizations. In *23rd Annual Symposium on Theoretical Aspects of Computer Science*, volume 3884 of *Lecture Notes in Computer Science*, pages 571–583, Marseille, France, 2006. Springer.
7. B. Doerr, T. Friedrich, C. Klein, and R. Osbild. Unbiased matrix rounding. In *10th Scandinavian Workshop on Algorithm Theory*, volume 4059 of *Lecture Notes in Computer Science*, pages 102–112, Riga, Latvia, 2006. Springer.
8. I. P. Fellegi. Controlled random rounding. *Survey Methodology*, 1:123–133, 1975.
9. P. Raghavan. Probabilistic construction of deterministic algorithms: Approximating packing integer programs. *J. Comput. Syst. Sci.*, 37:130–143, 1988.
10. J. Šíma. Table rounding problem. *Computers and Artificial Intelligence*, 18:175–189, 1999.
11. J. Spencer. Randomization, derandomization and antirandomization: Three games. *Theoretical Computer Science*, 131:415–429, 1994.
12. L. Willenborg and T. de Waal. *Elements of Statistical Disclosure Control*, volume 155 of *Lecture Notes in Statistics*. Springer, 2001.