Algorithmica (2018) 80:3943–3957
https://doi.org/10.1007/s00453-018-0492-8

Extreme Witnesses and Their Applications

Andrzej Lingas[1] · **Mia Persson**[2]

Received: 20 April 2016 / Accepted: 28 July 2018 / Published online: 6 August 2018

Abstract
We study the problem of computing the so called minimum and maximum witnesses for Boolean vector convolution. We also consider a generalization of the problem which is to determine for each positive value at a coordinate of the convolution vector, q smallest (largest) witnesses, where q is the minimum of a parameter k and the number of witnesses for this coordinate. We term this problem the smallest k-witness problem or the largest k-witness problem, respectively. We also study the corresponding smallest and largest k-witness problems for Boolean matrix product. First, we present an $\tilde{O}(n^{1.5}k^{0.5})$-time algorithm for the smallest or largest k-witness problem for the Boolean convolution of two n-dimensional vectors, where the notation $\tilde{O}(\ )$ suppresses polylogarithmic in n factors. In consequence, we obtain new upper time bounds on reporting positions of mismatches in potential string alignments and on computing restricted cases of the (min, +) vector convolution. Next, we present a fast (substantially subcubic in n and linear in k) algorithm for the smallest or largest k-witness problem for the Boolean matrix product of two $n \times n$ Boolean matrices. It yields fast algorithms for reporting k lightest (heaviest) triangles in a vertex-weighted graph.

Keywords Boolean vector convolution · Boolean matrix product · String matching · Witnesses · Minimum and maximum witnesses · Lightest triangles · Time complexity

A preliminary, short version of this paper has appeared in *Proceedings of the 9th International Conference on Combinatorial Optimization and Applications (COCOA 2015), Lecture Notes in Computer Science Volume 9486, Springer, Houston, TX, USA, December 18–20, 2015*.

✉ Andrzej Lingas
Andrzej.Lingas@cs.lth.se

Mia Persson
Mia.Persson@mah.se

[1] Department of Computer Science, Lund University, 22100 Lund, Sweden

[2] Department of Computer Science, Malmö University, 20506 Malmö, Sweden

1 Introduction

For a potential alignment of a pattern string with a text string over the same alphabet, a position in the alignment where the pattern symbol is different from the text symbol is a *witness* to the symbol *mismatch* while a position where the pattern and text symbol are equal is a witness to the symbol *match*.

Similarly, if A and B are two $n \times n$ Boolean matrices and C is their Boolean matrix product then for any entry $C[i, j] = 1$ of C, a witness is an index m such that $A[i, m] \wedge B[m, j] = 1$. The smallest (or, largest) possible witness is called the *minimum witness* (or, *maximum witness*, respectively).

The problems of finding "witnesses" have been extensively studied for several decades, at the beginning independently within stringology and graph algorithms relying on matrix computations. In string matching, witnesses for symbol mismatches or matches in potential alignments of two strings are sought [4,9,17] while in graph algorithms, witnesses for the Boolean matrix product are typically sought, originally in order to solve shortest path problems in graphs [2,3]. In both cases, highly non-trivial efficient algorithmic solutions have been presented [2–4,17].

Also in both areas, useful generalizations and/or specializations of the problems of finding witnesses have been studied. A natural generalization introduced for string matching in [17] is to request up to k witnesses instead of a single one. It has been efficiently solved by using concepts from group testing in [4] and conveyed to Boolean matrix product in [4,14]. A natural specialization is to request minimum or maximum witnesses. This specialization has been introduced and efficiently solved in [10] in the context of finding lowest common ancestors in directed acyclic graphs and it found many other applications since then (cf. [8,18,21]).

In analogy to witnesses for Boolean matrix product, if a and b are two n-dimensional Boolean vectors and c is their Boolean convolution then for any coordinate $c_i = 1$ of c, a *witness* is an index l such that $a_l \wedge b_{i-l} = 1$. In contrast to string matching and Boolean matrix product, the problem of computing the witnesses of Boolean vector convolution does not seem to be explicitly studied in the literature. On the other hand, Boolean vector convolution is very much related to string matching [12], and hence the algorithms for reporting witness or more generally up to k witnesses can be easily conveyed from stringology to Boolean vector convolution (see Proposition 3.1).

In this paper, we study the problem of computing minimum and maximum witnesses for Boolean vector convolution. We also consider a generalization of the problem which is to determine for each positive (value at a)[1] coordinate of the convolution vector, q smallest (largest) witnesses, where q is the minimum of a parameter k and the number of witnesses for this coordinate. We term this problem the smallest k-witness problem or the largest k-witness problem, respectively. We also study the corresponding generalization for Boolean matrix product.

Let $\omega(1, r, 1)$ denote the exponent of fast arithmetic multiplication of an $n \times n^r$ matrix by an $n^r \times n$ matrix. In particular, $\omega(1, 1, 1)$ denoted by ω is known to not exceed 2.373 [15,22]. Next, let the notation $\tilde{O}(\,)$ suppress polylogarithmic in n factors. Our main contributions are as follows:

[1] For brevity, we shall identify the i-th coordinate of a vector v with its value v_i in the continuation.

Table 1 Our upper time bounds for computing the (min, +) convolution of two n-dimensional integer vectors either with coordinates having a bounded number of different values, or decomposable into a number of non-decreasing or non-increasing subsequences, or just monotone subsequences

Vector a/vector b	c_b dif. values	m_b non-decr. subs.	m_b non-incr. subs.
c_a different values	$\tilde{O}(c_a c_b n)$	$\tilde{O}(c_a m_b n^{1.5})$	$\tilde{O}(c_a m_b n^{1.5})$
m_a non-decr. subs.	$\tilde{O}(m_a c_b n^{1.5})$	$\tilde{O}(m_a m_b n^{1.5})$	?
m_a non-incr. subs.	$\tilde{O}(m_a c_b n^{1.5})$	?	$\tilde{O}(m_a m_b n^{1.5})$
m_a mon. subs.	$\tilde{O}(m_a c_b n^{1.5})$	?	?
arbitrary	$\tilde{O}(c_b n^{1.844})$	?	?

- an $\tilde{O}(n^{1.5})$-time algorithm for reporting minimum and maximum witnesses for the Boolean convolution of two n-dimensional vectors, and more generally, an $\tilde{O}(n^{1.5}k^{0.5})$-time algorithm for the smallest or largest k-witness problem for the convolution;
- as corollaries, $\tilde{O}(n^{1.5}k^{0.5})$ time bounds for the smallest or largest k-witness problems in string matching;
- in part as corollaries, several upper time bounds on computing the (min, +) integer vector convolution in restricted cases, summarized in Table 1;
- an $O(n^{2+\lambda}k)$-time algorithm for the smallest or largest k-witness problem for the Boolean matrix product of two $n \times n$ Boolean matrices, where λ is a solution to the equation $\omega(1, \lambda, 1) = 1 + 2\lambda + \log_n k$;
- as a corollary, an $O(n^{2+\lambda}k)$ time bound for the problem of reporting for each edge of a vertex-weighted graph k lightest (heaviest) triangles containing it, where λ satisfies the aforementioned equation; also, an $O(\min\{n^{\omega}k + n^{2+o(1)}k, n^{2+\lambda}k\})$ time bound for the problem of reporting k lightest (heaviest) triangles in the input vertex-weighted graph.

2 Preliminaries

For two n-dimensional vectors $a = (a_0, \ldots, a_{n-1})$ and $b = (b_0, \ldots, b_{n-1})$ over a semi-ring $(\mathbb{U}, \oplus, \odot)$, their convolution over the semi-ring is a vector $c = (c_0, \ldots, c_{2n-2})$, where $c_i = \bigoplus_{l=\max\{i-n+1,0\}}^{\min\{i,n-1\}} a_l \odot b_{i-l}$ for $i = 0, \ldots, 2n-2$. Similarly, for a $p \times q$ matrix A and a $q \times r$ matrix B over the semi-ring, their matrix product over the semi-ring is a $p \times r$ matrix C such that $C[i, j] = \bigoplus_{m=1}^{q} A[i, m] \odot B[m, j]$ for $1 \le i \le p$ and $1 \le j \le r$. In particular, for the semi-rings $(\mathbb{Z}, +, \times)$, $(\mathbb{Z}, \min, +)$, $(\mathbb{Z}, \max, +)$, and $(\{0, 1\}, \vee, \wedge)$, we obtain the arithmetic, (min, +), (max, +), and the Boolean convolutions or matrix products, respectively.

We shall use the unit-cost RAM computational model [1] with computer word of length logarithmic in the maximum of the size of the input and the value of the largest input integer.

The following fact is well known (cf. [12]).

Fact 2.1 *Let p and q be two n-dimensional integer vectors. The arithmetic convolution of p and q can be computed in $\tilde{O}(n)$ time. Hence, also the Boolean convolution of two n-dimensional vectors can be computed in $\tilde{O}(n)$ time.*

For a sequence S of integers, we shall denote the minimum number of monotone subsequences into which S can be decomposed by $mon(S)$.

Fact 2.2 [13,23] *A sequence of n integers can be decomposed into $O(mon(S)\log n)$ monotone subsequences in $O(n^{1.5}\log n)$ time.*

Fact 2.3 (see Theorem 10 in [5]) *The problem of computing the convolution of two n-dimensional vectors over a semi-ring can be reduced to computing $O(\sqrt{n})$ products of two $O(\sqrt{n}) \times O(\sqrt{n})$ matrices over the semi-ring. Importantly, the matrices can be constructed in $O(n^{1.5})$ time in total and their entries are appropriately filled with the coordinates of the vectors.*

Fact 2.4 (Theorem 3.2 in [7]) *Let A and B be two $n \times n$ integer matrices where the entries of A range over at most c different integers. The* $(\min, +)$ *matrix product of A and B can be computed in $O(cn^{2.688})$ time.*

Fact 2.5 [11] *A lightest (heaviest) triangle in an undirected vertex weighted graph on n vertices can be found in $O(n^{\omega} + n^{2+o(1)})$ time.*

3 Extreme Witnesses for Boolean Convolution

Let $c = (c_0, \ldots, c_{2n-2})$ be the Boolean convolution of two n-dimensional Boolean vectors a and b. A *witness* of $c_i = 1$ is any $l \in [\max\{i - n + 1, 0\}, \min\{i, n - 1\}]$ such that $a_l \wedge b_{i-l} = 1$. A *minimum witness* (or *maximum witness*) of $c_i = 1$ is the smallest (or, the largest, respectively) witness of c_i. The *witnesses problem* (or *minimum witness problem*, or *maximum witness problem*) for the Boolean convolution of two n-dimensional Boolean vectors is to determine witnesses (or, the minimum witnesses or the maximum witnesses, respectively) for all non-zero coordinates of the Boolean convolution of the vectors. The *k-witness problem* (or, *the smallest k-witness problem* or *the largest k-witness problem*) for the Boolean convolution of two n-dimensional Boolean vectors is to determine for each non-zero coordinate of the convolution q witnesses (or, q smallest witnesses or q largest witnesses, respectively), where q is the minimum of k and the number of witnesses for this coordinate.

The Boolean vector convolution is very much related to string matching problems [12]. The corresponding problems of reporting a symbol mismatch or match, or up to k such mismatches or matches for each potential alignments of the pattern with the text have been studied in the so called non-standard stringology [4,17]. Also, the focus of this paper is on extreme witnesses. For these reasons and on the other hand, for the completeness sake, we just state a proposition and its generalization on standard witnesses for Boolean vector convolution that can be obtained analogously as the well known corresponding facts on string matching or Boolean matrix product.

Proposition 3.1 (Analogous to [3]) *The witnesses problem for Boolean convolution of two n-dimensional vectors can be solved in $\tilde{O}(n)$ time.*

Proof sketch. The witnesses for the Boolean convolution c of two n-dimensional vectors a and b can be computed analogously as the witnesses for the Boolean matrix product [3]. The first observation is that for all coordinates of c that have a single witness, their witnesses can be obtained by computing the arithmetic convolution of a with the vector b' resulting from replacing each 1 in b with the number of the respective coordinate. The next idea is to dilute the other vector b gradually so the number of witnesses for each positive coordinate of c decreases finally to zero but in most cases passing through 1 first. For instance, if c_i has l witnesses and in each phase each coordinate of b is set to 0 with probability $\frac{1}{2}$ then after a logarithmic number of such phases there is a positive probability that exactly one witness will remain. By iterating the process a logarithmic number of times witnesses for all positive coordinates of c can be determined with high probability.

In order to remove the randomness, we can use small c-wise ϵ-bias sample spaces analogously as Alon and Naor in their deterministic algorithm for witnesses of Boolean matrix product [3].

The algorithm, its analysis and derandomization are totally analogous to those of the algorithm of Alon and Naor for witnesses of Boolean matrix product [3]. We refer the reader for the technical details to their paper. It is sufficient to replace matrices with vectors, entries with coordinates and Boolean matrix product with Boolean vector convolution in their proof. □

Following [4] and [14], one can also generalize Proposition 1 to include an algorithmic solution to the k-witness problem for Boolean convolution of two n-dimensional vectors in $\tilde{O}(nk)$ time.

With a moderate technical effort, the minimum or maximum witness problem for Boolean convolution could be solved by combining the known $O(n^{2.575})$-time algorithm for the corresponding problem of minimum or maximum witnesses of Boolean matrix product [10] with the known reduction of vector convolution over an arbitrary semi-ring to matrix product over the semi-ring described in Fact 2.3 [5]. The combination results in an $O(n^{1.787})$-time solution to the extreme witness problem for Boolean convolution. We shall show that a substantially more efficient solution can be obtained directly.

Theorem 3.2 *The minimum witness problem (maximum witness problem, respectively) for Boolean convolution of two n-dimensional vectors can be solved in $\tilde{O}(n^{1.5})$ time.*

Proof Let a and b be two n-dimensional vectors. Let r be an integer parameter between 1 and n. For $p = 1, \ldots, \lceil n/r \rceil$, let a^p be the Boolean n-dimensional vector resulting from setting to zero all coordinates of a with indices not exceeding $(p-1)r$ and those with indices greater than pr. We compute, for each $p = 1, \ldots, \lceil n/r \rceil$, the Boolean convolution c^p of a^p and b. Next, for each $i = 0, \ldots, 2n-2$, we determine the smallest p such that $c_i^p = 1$. Then, if such a p exists, we determine the interval of the implicants $a_l \wedge b_{i-l}$ of c_i^p that potentially can have a non-zero value, i.e., where $l \in ((p-1)r, pr]$, and return the smallest l in the interval for which $a_l \wedge b_{i-l} = 1$. The $\lceil n/r \rceil$ computations of Boolean convolutions c^p takes $\tilde{O}(n^2/r)$ time. The total time taken by the determination of the smallest p is $O(n \times n/r)$. To determine the smallest l for a given i and p requires examining the value of $O(r)$ implicants and

hence it takes $O(nr)$ time in total. By setting $r = \lceil \sqrt{n} \rceil$, we obtain the claimed time complexity. □

The method of Theorem 3.2 can be generalized to include the smallest k-witness problem and the largest k-witness problem.

Theorem 3.3 *The smallest k-witness problem as well as the largest k-witness problem for Boolean convolution of two n-dimensional vectors can be solved in $\tilde{O}(n^{1.5}k^{0.5})$ time.*

Proof Let a and b be two input n-dimensional vectors. Let r be an integer parameter between 1 and n. Analogously as in the proof of Theorem 3.2, for $p = 1, \ldots, \lceil n/r \rceil$, we let a^p denote the Boolean n-dimensional vector resulting from setting to zero all coordinates of a with indices not exceeding $(p-1)r$ and those with indices greater than pr. Next, we compute for each $p = 1, \ldots, \lceil n/r \rceil$, the arithmetic convolution w^p of a^l and b by interpreting these vectors as $0-1$ ones. The arithmetic convolutions provide us with the number of witnesses in each interval $((p-1)r, pr]$ for each coordinate c_i of the Boolean convolution c of a and b. Their coordinate-wise sum provides us with the total number of witnesses for each coordinate of c. In order to solve the smallest k-witness problem, for $p = 1, \ldots, \lceil n/r \rceil$, and for $i = 0, \ldots, 2n-2$, whenever $w_i^p > 0$ and the number of witnesses for c_i found so far is less than the minimum of k and the number of witnesses of c_i, we search through the interval $((p-1)r, pr]$ from the left to the right for further witnesses. For details see the algorithm depicted in Fig. 1. In the worst case, for each $i = 0, \ldots, 2n-2$, we need to search through k of such intervals. The total cost of the searches becomes $O(n \times \frac{n}{r} + n \times k \times r)$, see lines 15–19 in the algorithm depicted in Fig. 1. On the other hand, the $\lceil n/r \rceil$ computations of the arithmetic convolutions w^p takes $\tilde{O}(n^2/r)$ time. By setting $r = \lceil \sqrt{\frac{n}{k}} \rceil$, we obtain the claimed time complexity for the smallest k-witness problem.

The largest k-witness problem can be solved analogously in the same asymptotic time by considering the intervals in the opposite order and searching them from the right to the left instead. □

3.1 String Matching

Fisher and Patterson showed already in 1974 [12] that several string matching problems can be efficiently reduced to Boolean vector convolution.

Suppose we are given two strings $\tau = \tau_{m-1}\tau_{m-2}...\tau_0$ and $\rho = \rho_0\rho_1...\rho_{n-1}$, where $m < n$, over a finite alphabet Σ. Following [12], for $\gamma \in \Sigma$, let $H_\gamma(\,)$ be a function from Σ to { true, false } such that $H_\gamma(x) = true$ if and only if $x = \gamma$. If $i + m \leq n$, the question of whether $\tau_{m-1}\tau_{m-2}...\tau_0$ matches $\rho_i\rho_{i+1}...\rho_{i+m-1}$ is equivalent to a conjunction of the negations of terms $\bigvee_{l=0}^{m-1} H_\alpha(\rho_{i+l}) \wedge H_\beta(\tau_{m-1-l})$, where $\alpha, \beta \in \Sigma$ and $\alpha \neq \beta$. Note that whenever such a term is true, the matching cannot take place as at some position α clashes with β. In this way, the standard string matching problem for τ and ρ easily reduces to $O(|\Sigma|^2)$ Boolean convolutions of two Boolean vectors of length at most n.

Input: two n-dimensional Boolean vectors $a = (a_0, ..., a_{n-1})$ and $b = (b_0, ..., b_{n-1})$, and an integer parameter $k \in [1, n]$.
Output: for each coordinate c_i of the Boolean convolution $c = (c_0, ..., c_{2n-2})$ of a and b, q smallest witnesses of c_i, where q is the minimum of k and the number of witnesses of c_i.

for $i = 0$ **to** $2n - 2$ **do**
  $w_i \leftarrow 0$
  $witset(c_i) \leftarrow \emptyset$
end for
$r \leftarrow \lceil \sqrt{\frac{n}{k}} \rceil$
for $p = 1$ **to** $\lceil n/r \rceil$ **do**
  set a^p to an n-dimensional Boolean vector obtained from a by zeroing all coordinates of a outside the interval $((p-1)r, pr]$
end for
for $p = 1$ **to** $\lceil n/r \rceil$ **do**
  compute the arithmetic convolution w^p of a^p and b treated as $0-1$ vectors
  for $i = 0$ **to** $2n - 2$ **do**
    $w_i \leftarrow w_i + w_i^p$
  end for
end for
for $p = 1$ **to** $\lceil n/r \rceil$ **do**
  for $i = 0$ **to** $2n - 2$ **do**
    if $\#witset(c_i) < \min\{w_i, k\} \wedge w_i^p \geq 1$ **then** extend $witset(c_i)$ by $\min\{w_i^p, \min\{w_i, k\} - \#witset(c_i)\}$ smallest witnesses of c_i in the interval $((p-1)r, pr]$.
  end for
end for
$witset \leftarrow (witset(c_0), ..., witset((c_{2n-2}))$
return $witset$

Fig. 1 An algorithm for the smallest k-witness problem for the Boolean convolution of two n-dimensional vectors a and b

Observe now that witnesses for the aforementioned Boolean convolutions yield positions of the clashes, in other words, symbol mismatches. If we modify the terms to $\bigvee_{l=0}^{m-1} H_\alpha(\rho_{i+l}) \wedge H_\alpha(\tau_{m-1-l})$, for $\alpha \in \Sigma$, the witnesses for the $O(|\Sigma|)$ Boolean convolutions yield positions of two sided matches with $\alpha \in \Sigma$. Hence, we obtain the following theorem as a corollary from Theorem 3.3.

Theorem 3.4 *Consider the string matching problem for a text string of length n and a pattern string of length $m < n$, both over a finite alphabet. For each alignment of the pattern with the text, we can provide locations of the k earliest symbol mismatches and/or the k earliest symbol matches as well as locations of the k latest symbol mismatches and the k latest symbol matches in the alignments in $\tilde{O}(n^{1.5}k^{0.5})$ time in total. In particular, we can also provide positions of the earliest and/or latest two-side symbol matches with a given alphabet symbol (cf. ones problem in [17]) in the alignments in $\tilde{O}(n^{1.5}k^{0.5})$ time in total.*

3.2 (min, +) Convolution

Our original motivation has been an extension of the $O(n^{1.859})$-time algorithm due to Chan and Levenstein for the (min, +) convolution of two n-dimensional vectors with integer coordinates of size $O(n)$ forming monotone sequences [6] to include

Input: two n-dimensional vectors $a = (a_0, ..., a_{n-1})$ and $b = (b_0, ..., b_{n-1})$ with integer coordinates such that the coordinates of a range over c_a different values and the sequence of consecutive coordinates of b is decomposable into m_b monotone subsequences.
Output: the $(\min, +)$ convolution $c = (c_0, ..., c_{2n-2})$ of a and b
1: Decompose the sequence $a_0, ..., a_{n-1}$ into a minimum number of constant subsequences $a^i = a^i_1, ..., a^i_{l_i}$
2: **for** each a^i **do**
3: form an n-dimensional Boolean vector $char(a^i)$ with n coordinates indicating with ones the coordinates of a covered by a^i
4: **end for**
5: Decompose the sequence $b_0, ..., b_{n-1}$ into a number of monotone subsequences $b^j = b^j_1, ..., b^j_{l_p}$ within $O(\log n)$ of the minimum
6: **for** each b^j **do**
7: form an n-dimensional Boolean vector $char(b^j)$ with n coordinates indicating with ones the coordinates of b covered by b^j
8: **end for**
9: **for** each pair a^i and b^j **do**
10: **if** b^j is non-decreasing **then** compute the minimum witnesses $wit(d_0)$,...,$wit(d_{2n-2})$ of the Boolean convolution $d = (d_0, ..., d_{2n-2})$ of $char(a^i)$ and $char(b^j)$
11: **if** b^j is non-increasing **then** compute the maximum witnesses $wit(d_0)$,...,$wit(d_{2n-2})$ of the Boolean convolution $d = (d_0, ..., d_{2n-2})$ of $char(a^i)$ and $char(b^j)$
12: **for** $k = 0$ **to** $2n - 2$ **do**
13: **if** $d_k \neq 0$ **then** $c_k \leftarrow \min\{a^i_{wit(d_k)} + b^j_{k-wit(d_k)}, c_k\}$
14: **end for**
15: **end for**
16: $c \leftarrow (c_0, ..., c_{2n-2})$
17: **return** c

Fig. 2 An algorithm for computing the $(\min, +)$ convolution c of two n-dimensional integer vectors a and b, where the coordinates of a range over c_a different values and the sequence of consecutive coordinates of b is decomposable into m_b monotone subsequences

the case where the vectors are decomposable into relatively few monotone subsequences. The major difficulty here is that a completion of the subsequences to full monotone sequences can affect the result. Roughly, we can avoid this difficulty when the coordinates of each of the vectors range over relatively few different values or all the subsequences are simultaneously either non-decreasing or non-increasing (see Table 1). The idea is to use our algorithm for minimum and maximum witnesses of Boolean convolution.

The correctness of the algorithm depicted in Fig. 2 relies on the following straightforward lemma.

Lemma 3.5 *In the algorithm depicted in Fig. 2, the following equivalence holds: $d_k \neq 0$ in line 13 if and only if $\min\{a_l + b_m | l + m = k \wedge a_l \in a^i \wedge b_m \in b^j\}$ is equal to the first argument of the minimum in this line.*

Theorem 3.6 *Let a and b be two n-dimensional integer vectors such that the coordinates of a range over at most c_a different values while the sequence of the consecutive coordinates of b can be decomposed into m_b monotone subsequences. The algorithm depicted in Fig. 2 computes their $(\min, +)$ convolution in $\tilde{O}(c_a m_b n^{1.5})$ steps.*

Proof By Lemma 3.5 and line 13 in the algorithm, none of the coordinates of the output vector has a lower value than the corresponding coordinate of the $(\min, +)$

Input: two n-dimensional vectors $a = (a_0, ..., a_{n-1})$ and $b = (b_0, ..., b_{n-1})$ with integer coordinates and their decompositions into m_a and m_b subsequences $a^i = a^i_1, ..., a^i_{l_i}$ and $b^j = b^j_1, ..., b^j_{l_p}$ respectively that are either all non-decreasing or all non-increasing.

Output: the $(\min, +)$ convolution $c = (c_0, ..., c_{2n-2})$ of a and b

1: **for** each a^i **do**
2: form a Boolean vector $char(a^i)$ with n coordinates indicating with ones the coordinates of a covered by a^i
3: **end for**
4: **for** each b^j **do**
5: form a Boolean vector $char(b^j)$ with n coordinates indicating with ones the coordinates of b covered by b^j
6: **end for**
7: **for** each pair a^i and b^j **do**
8: **if** the subsequences are non-decreasing **then** compute the minimum witnesses $wit(d_0)$,...,$wit(d_{2n-2})$ of the Boolean convolution d of $char(a^i)$ and $char(b^j)$
9: **if** the subsequences are non-increasing **then** compute the maximum witnesses $wit(d_0)$,...,$wit(d_{2n-2})$ of the Boolean convolution d of $char(a^i)$ and $char(b^j)$
10: **for** $k = 0$ **to** $2n - 2$ **do**
11: **if** $d_k \neq 0$ **then** $c_k \leftarrow \min\{a^i_{wit(d_k)} + b^j_{k-wit(d_k)}, c_k\}$
12: **end for**
13: **end for**
14: $c \leftarrow (c_0, ..., c_{2n-2})$
15: **return** c

Fig. 3 An algorithm for computing the $(\min, +)$ convolution c of two n-dimensional integer vectors a and b given with their decompositions into m_a and m_b subsequences that are either all non-decreasing or all non-increasing

convolution of a and b. Conversely, if the k-th coordinate of the $(\min, +)$ convolution of a and b equals $a_l + b_m$, where $l + m = k$, then there exists i, j such that $a_l \in a^i$ and $b_m \in b^j$. Hence again by Lemma 3.5 and line 13 in the algorithm, the k-th coordinate in the output vector has value not larger than the k-th coordinate of the $(\min, +)$ convolution of a and b.

The decomposition of the vector a into c_a constant subsequences in line 1 trivially takes $O(n)$ time. Next, the decomposition of the vector b into $\tilde{O}(m_a)$ monotone subsequences in line 5 takes $O(n^{1.5} \log n)$ time by Fact 2.2. The forming of the vectors $char(a^i)$ in lines 2–3 and $char(b^j)$ in lines 6–7 take $\tilde{O}(c_a n + m_b n)$ time in total. The $\tilde{O}(c_a m_b)$ computations of the minimum and maximum witnesses of the Boolean convolution d in lines 9–11 take $\tilde{O}(c_a m_b n^{1.5})$ time in total by Theorem 3.2. Finally, the line 13 is executed $\tilde{O}(c_a m_b n)$ times. The bound $\tilde{O}(c_a m_b n^{1.5})$ follows.

If we are given decompositions of the two input n-dimensional vectors a and b into monotone subsequences that are either all non-decreasing or all non-increasing then we can use the algorithm depicted in Fig. 3 which is analogous to that depicted in Fig. 2, in order to compute the $(\min, +)$ convolution of a and b. Thus, first for each subsequence a^i of a and each subsequence b^j of b, we compute the Boolean vectors $char(a^i)$ and $char(b^j)$ indicating with ones the coordinates of a or b covered by a^i or b^j, respectively. Next, depending if the subsequences are non-decreasing or non-increasing, for each pair of such subsequences a^i and b^j, we compute the minimum witnesses of the Boolean convolution of $char(a^i)$ and $char(b^j)$ or the maximum witnesses of this convolution, respectively. We use the extreme witnesses to update

Input: two n-dimensional integer vectors $a = (a_0, ..., a_{n-1})$ and $b = (b_0, ..., b_{n-1})$ whose coordinates range over at most c_a or c_b different values, respectively.
Output: the $(\min, +)$ convolution $c = (c_0, ..., c_{2n-2})$ of a and b.
1: Decompose the sequence $a_0, ..., a_{n-1}$ into a minimum number of constant subsequences $a^i = a^i{}_1, ..., a^i{}_{l_i}$
2: **for** each a^i **do**
3: form an $0-1$ vector $char(a^i)$ with n coordinates indicating with ones the coordinates of a covered by a^i
4: **end for**
5: Decompose the sequence $b_0, ..., b_{n-1}$ into a minimum number of constant subsequences $b^j = b^j_1, ..., b^j{}_{l_i}$
6: **for** each b^j **do**
7: form an $0-1$ vector $char(b^j)$ with n coordinates indicating with ones the coordinates of b covered by b^j
8: **end for**
9: **for** each pair a^i and b^j **do**
10: compute the Boolean convolution d of $char(a^i)$ and $char(b^j)$
11: **for** $k = 0$ **to** $2n - 2$ **do**
12: **if** $d_k \neq 0$ **then** $c_k \leftarrow \min\{a^i_1 + b^j_1, c_k\}$
13: **end for**
14: **end for**
15: $c \leftarrow (c_0, ..., c_{2n-2})$
16: **return** c

Fig. 4 An algorithm for computing the $(\min, +)$ convolution c of two n-dimensional integer vectors a and b whose coordinates range over at most c_a or c_b different values, respectively

the current coordinates of the computed $(\min, +)$ convolution analogously as in the algorithm depicted in Fig. 2. Hence, we obtain the following theorem.

Theorem 3.7 *Let a and b be two n-dimensional integer vectors given with the decompositions of the sequences of their consecutive coordinates into m_a and m_b monotone subsequences respectively such that all the subsequences are either non-decreasing or non-increasing. The algorithm depicted in Fig. 3 computes the* $(\min, +)$ *convolution of a and b in $\tilde{O}(m_a m_b n^{1.5})$ time.*

Proof The proof of the correctness of the algorithm depicted in Fig. 3 is analogous to that of the correctness of the algorithm depicted in Fig. 2. The time complexity analysis of the former algorithm is also similar to that of the latter algorithm. The main difference is that the decompositions of a and b into subsequences are given and that the $O(n^{1.5})$-time algorithm for minimum or maximum witnesses of Boolean convolution is run $m_a m_b$ times instead of $c_a m_b$ times. □

By combining Fact 2.3 with Fact 2.4, we also obtain the following bound.

Theorem 3.8 *Let a and b be two n-dimensional integer vectors such that the coordinates of a range over at most c_a different values. The* $(\min, +)$ *convolution of a and b can be computed in $\tilde{O}(c_a n^{1.844})$ time.*

We can also consider the problem of computing the $(\min, +)$ integer vector convolution of the input vectors a and b, when their coordinates range over c_a and c_b different integers, respectively. We can use the algorithm depicted in Fig. 4, analogous to that depicted in Fig. 2. The first difference is that the subsequences b^j on the side of

b are also constant. It follows that for any pair of such constant subsequences a^i and b^j, the value of the sum of any element from a^i with any element from b^j is constant and it can be trivially computed as $a^i_1 + b^j_1$ a priori. For this reason, it is sufficient to compute the Boolean convolution d of $char(a^i)$ and $char(b^j)$ for each pair a^i and b^j. Then, for any non-zero coordinate of d, we need to update the corresponding coordinate of the computed $(\min, +)$ convolution of a and b by taking the minimum of the coordinate and $a^i_1 + b^j_1$. By Fact 2.1, we obtain the following theorem.

Theorem 3.9 *Let a and b be two n-dimensional integer vectors such that their coordinates range over at most c_a or c_b different values, respectively. The algorithm depicted in Fig. 4 computes the $(\min, +)$ convolution of a and b in $\tilde{O}(c_a c_b n)$ time.*

Proof The algorithm depicted in Fig. 4 can be easily implemented in $\tilde{O}(c_a c_b n)$ time by running $c_a c_b$ times the known $\tilde{O}(n)$-time algorithm for Boolean convolution of two n-dimensional Boolean vectors, see Fact 2.1. □

4 Extreme Witnesses for Boolean Matrix Product

For two $n \times n$ Boolean matrices A and B, a witness of a $C[i, j]$ entry of the Boolean matrix product of A and B is any index m such that $A[i, m] \wedge B[m, j] = 1$. Next, the minimum witness and maximum witness for an entry of C as well as the witness problem, the minimum and maximum witness problems, the k-witness problem, and the smallest k-witness and largest k-witness problems for Boolean matrix product of A and B are defined analogously as those for Boolean vector convolution.

In this section, we shall present a generalization of the algorithm for minimum and maximum witnesses for Boolean matrix product from [10] to include the smallest and largest k-witness problems.

Let ℓ be a positive integer smaller than n. We may assume w.l.o.g. that n is divisible by ℓ. Partition the matrix A into $n \times \ell$ sub-matrices A_p and the matrix B into $\ell \times n$ sub-matrices B_p, such that $1 \leq p \leq n/\ell$, and the sub-matrix A_p covers the columns $(p-1)\,\ell+1$ through $p\,\ell$ of A whereas the sub-matrix B_p covers the rows $(p-1)\,\ell+1$ through $p\,\ell$ of B.

For $p = 1, \ldots, n/\ell$, let W_p be the arithmetic product of A_p and B_p treated as $0-1$ matrices. On the other hand, let C denote the Boolean matrix product of A and B. Then, $W_p[i, j] = q$ if and only if there are exactly q witnesses of $C[i, j]$ in the interval $((p-1)\,\ell, p\,\ell]$. Consequently, the total number of witnesses of $C[i, j]$ is given by $\sum_{p=1}^{n/\ell} W_p[i, j]$. Therefore, the following lemma follows.

Lemma 4.1 *Suppose that a $C[i, j]$ entry of the Boolean product C of A and B is positive. Let q be the minimum of k and the total number of witnesses of $C[i, j]$. Next, let p' be the minimum value of p such that $\sum_{u=1}^{p} W_u[i, j]$ is not less than q. The smallest q witnesses of $C[i, j]$ belong to the interval $[1, p'\,\ell]$.*

By this lemma, after computing all the matrix products $W_p = A_p \cdot B_p, 1 \leq p \leq n/\ell$, we need $O(n/\ell + k\ell)$ time per positive entry of C to find up to k smallest witnesses:

Input: two $n \times n$ Boolean matrices A and B, and integer parameters $\ell, k \in [1, n]$, where n is assumed to be divisible by ℓ.
Output: for each entry $C[i, j]$ of the Boolean matrix product C of A and B, q smallest witnesses of $C[i, j]$, where q is the minimum of k and the number of witnesses of $C[i, j]$.

for $i = 1$ **to** n **do**
  for $j = 1$ **to** n **do**
    $W[i, j] \leftarrow 0$
    $witset(C[i, j]) \leftarrow \emptyset$
  end for
end for
for $p = 1$ **to** n/ℓ **do**
  set A_p to an $n \times n$ Boolean matrix that is a restriction of A to columns with indices in $((p-1)\ell, p\ell]$
end for
for $p = 1$ **to** n/ℓ **do**
  set B_p to an $n \times n$ Boolean matrix that is a restriction of B to rows with indices in $((p-1)\ell, p\ell]$
end for
for $p = 1$ **to** n/ℓ **do**
  compute the arithmetic matrix product W_p of A_p and B_p treated as $0-1$ matrices
  for $i = 1$ **to** n **do**
    for $j = 1$ **to** n **do**
      $W[i, j] \leftarrow W[i, j] + W_p[i, j]$
    end for
  end for
end for
for $p = 1$ **to** n/ℓ **do**
  for $i = 1$ **to** n **do**
    for $j = 1$ **to** n **do**
      if $\#witset(C[i, j]) < \min\{W[i, j], k\} \wedge W_p[i, j] \geq 1$ **then** extend $witset(C[i, j])$ by $\min\{W_p[i, j], \min\{W[i, j], k\} - \#witset(C[i, j])\}$ smallest witnesses of $C[i, j]$ in the interval $((p-1)\ell, p\ell]$
    end for
  end for
end for
for $i = 1$ **to** n **do**
  for $j = 1$ **to** n **do**
    return $witset(C[i, j])$
  end for
end for

Fig. 5 An algorithm for the smallest k-witness problem for the Boolean matrix product of two $n \times n$ Boolean matrices A and B

$O(n/\ell)$ time to determine p' and then $O(k\ell)$ time to locate the up to k smallest witnesses. See Fig. 5 for our algorithm for the smallest k-witness problem.

Recall that $\omega(1, r, 1)$ denotes the exponent of the multiplication of an $n \times n^r$ matrix by an $n^r \times n$ matrix. It follows that the total time taken by our algorithm for the smallest k-witness problem is

$$O((n/\ell) \cdot n^{\omega(1,\log_n \ell,1)} + n^3/\ell + n^2 k\ell) \ .$$

By setting r to $\log_n \ell$ and z to $\log_n k$, our upper bound transforms to $O(n^{1-r+\omega(1,r,1)} + n^{3-r} + n^{2+r+z})$. Note that by assuming $r \geq \frac{1}{2} - \frac{z}{2}$, we can get rid of the additive n^{3-r} term. See Fig. 5 in [24] for the graph of the function $1 - r + \omega(1, r, 1)$. By solving

the equation $1 - \lambda + \omega(1, \lambda, 1) = 2 + z + \lambda$ implying $\lambda \geq \frac{1}{2} - \frac{z}{2}$ by $\omega(1, \lambda, 1) \geq 2$, we obtain our main result.

Theorem 4.2 *Let λ be such that $\omega(1, \lambda, 1) = 1 + 2\lambda + \log_n k$. The smallest k-witness problem as well as the largest k-witness problem for the Boolean matrix product of two $n \times n$ Boolean matrices can be solved in $O(n^{2+\lambda}k)$ time.*

Le Gall has recently substantially improved upper time bounds on rectangular matrix multiplication in [16]. In consequence, he could show that for the equation $\omega(1, \mu, 1) = 1 + 2\mu$, $\mu < 0.5302$. This in particular improves the upper time bound for the minimum and maximum witness problems from $O(n^{2.575})$ to $O(n^{2.5302})$. It follows that for $k \gg 1$, λ in Theorem 4.2 is substantially smaller than 0.5302.

4.1 Lightest Triangles

By generalizing the reduction of the problem of reporting for each edge of a vertex-weighted graph a heaviest triangle containing it to the maximum witness problem for Boolean matrix product from [21] to include reporting k heaviest triangles and the largest k-witness problem, we obtain the following theorem as a corollary from Theorem 4.2.

Theorem 4.3 *Let G be an undirected vertex weighted graph on n vertices and let k be a natural number not exceeding n. Next, let λ be such that $\omega(1, \lambda, 1) = 1 + 2\lambda + \log_n k$. We can list for each edge $\{u, v\}$ of G, q_e lightest (heaviest) triangles $\{u, v, w\}$ in G, where q_e is the minimum of k and the number of triangles $\{u, v, w\}$ in G, in $O(n^{2+\lambda}k)$ time.*

Proof Number the vertices of G in non-decreasing vertex-weight order. Next, solve the smallest (largest) k-witness problem for the Boolean matrix product C of the adjacency matrix of G with itself. For each edge $e = \{i, j\}$ of G, the up to k smallest (or, largest) witnesses of $C[i, j]$ yield the q_e lightest (or, heaviest, respectively) triangles in G including e. Theorem 4.2 yields the claimed upper bound. □

As for the problem of finding k lightest (heaviest) triangles in a vertex-weighted graph, iterating the $O(n^\omega + n^{2+o(1)})$-time algorithm for finding a lightest or heaviest triangle described in Fact 2.5 seems to be a better choice for up to moderate values of k. Before each next iteration, we remove the three vertices of the lastly reported triangle. After k iterations, we stop and find among the reported triangles and no more than $3(k-1)n^2$ other triangles incident to the removed vertices, the k lightest (heaviest) triangles if possible. The method takes $O(n^\omega k + n^{2+o(1)}k + n^2 k)$, i.e., $O(n^\omega k + n^{2+o(1)}k)$ time.

Theorem 4.4 *Let G be an undirected vertex weighted graph on n vertices and let k be a natural number not exceeding n. Next, let λ be such that $\omega(1, \lambda, 1) = 1 + 2\lambda + \log_n k$. We can list q lightest (heaviest) triangles in G, where q is the minimum of k and the number of triangles in G, in $O(\min\{n^\omega k + n^{2+o(1)}k, n^{2+\lambda}k\})$ time.*

Finding or detecting triangles of extreme weight in vertex-weighted graphs has a number of applications . First of all, it can be used to solve the corresponding general problem of finding or detecting subgraphs or induced subgraphs of extreme weigh [11,20,21]. Vassilevska and Williams list also two other applications in [19]: a general variant of the 3-SUM problem and a general buyer-seller problem in computational economy.

5 Final Remarks

It is an interesting open problem if any of our upper time bounds on minimum and maximum witnesses for Boolean vector convolution and the extreme k-witness problems both for Boolean vector convolution and Boolean matrix product can be substantially improved? Note here that so far the $O(n^{2+\lambda})$ time bound (where $\omega(1, \lambda, 1) = 1 + 2\lambda$) on minimum and maximum witnesses of Boolean matrix product established one decade ago [10] couldn't be improved (see also [8]).

The problems of Boolean vector convolution and Boolean matrix product seem to be similar but there are some substantial differences between them. The former problem admits almost a linear in the input size algorithm while for the latter problem the current upper time bound is substantially non-linear [15,22]. There is a moderately efficient reduction of vector convolution to matrix product described in Fact 2.3 while such a reverse reduction is not known. Our upper time bounds for minimum and maximum witnesses of Boolean vector convolution show that a direct approach to Boolean vector convolution can yield better upper time bounds than those obtained by conveying known upper time bounds for the witness problems for Boolean matrix product via Fact 2.3 to those corresponding for Boolean vector convolution.

The extreme k-witness problems for Boolean matrix product presumably admit several other applications often corresponding to generalizations of the applications for minimum and maximum witnesses of Boolean matrix product [18,21] and/or the applications of the k-witness problem for Boolean matrix product [4], e.g., the all-pairs k-bottleneck paths.

Finally, a potentially interesting direction for further research is to consider approximation variants of the extreme witnesses problems and the $(\min, +)$ vector convolution.

Acknowledgements We thank Mirosław Kowaluk and anonymous reviewers of the preliminary version of this paper for valuable comments. This research has been supported in part by Swedish Research Council Grant 621-2011-6179.

References

1. Aho, A.V., Hopcroft, J.E., Ullman, J.: The Design and Analysis of Computer Algorithms. Addison-Wesley Publishing Company, Reading (1974)
2. Alon, N., Galil, Z., Margalit, O., Naor, M.: Witnesses for Boolean matrix multiplication and for shortest paths. In: Proceedings of the 33rd Symposium on Foundations of Computer Science (FOCS), pp. 417–426 (1992)
3. Alon, N., Naor, M.: Derandomization, witnesses for Boolean matrix multiplication and construction of perfect hash functions. Algorithmica **16**, 434–449 (1996)
4. Aumman, N., Levenstein, M., Levenstein, N., Tsur, D.: Finding witnesses by peeling. In: Proceedings of the Combinatorial Pattern Matching (CPM). LNCS, vol. 4580, pp. 28–39. Springer (2007)
5. Bremner, D., Chan, T.M., Demaine, E.D., Erickson, J., Hurtado, F., Iacono, J., Langerman, S., Patrascu, M., Taslakian, P.: Necklaces, convolutions and X + Y. Algorithmica **69**, 294–314 (2014)
6. Chan, T.M., Lewenstein, M.: Clustered integer 3SUM vis additive combinatorics. In: Proceedings of the 47th ACM Symposium on Theory of Computing (STOC 2015)
7. Chan, T.M.: More algorithms for all-pairs shortest paths in weighted graphs. SIAM J. Comput. **39**(5), 2025–2089 (2010)
8. Cohen, K., Yuster, R.: On minimum witnesses for Boolean matrix multiplication. Algorithmica **69**(2), 431–442 (2014)
9. Crochemore, M., Rytter, W.: Text Algorithms. Oxford University Press, New York (1994)
10. Czumaj, A., Kowaluk, M., Lingas, A.: Faster algorithms for finding lowest common ancestors in directed acyclic graphs. In the special ICALP 2005 issue of Theoretical Computer Science **380**(1–2), 37–46 (2007)
11. Czumaj, A., Lingas, A.: Finding a heaviest vertex-weighted triangle is not harder than matrix multiplication. SIAM J. Comput. **39**(2), 431–444 (2009)
12. Fisher, M.J., Paterson, M.S.: String-matching and other products. In: Proceedings of the 7th SIAM-AMS Complexity of Computation, pp. 113–125 (1974)
13. Fomin, F.V., Kratsch, D., Novelli, J.: Approximating minimum cocolorings. Inf. Process. Lett. **84**, 285–290 (2002)
14. Gasieniec, L., Kowaluk, M., Lingas, A.: Faster multi-witnesses for Boolean matrix product. Inf. Process. Lett. **109**, 242–247 (2009)
15. Le Gall, F.: Powers of tensors and fast matrix multiplication. In: Proceedings of the 39th International Symposium on Symbolic and Algebraic Computation, pp. 296–303 (2014)
16. Le Gall, F.: Faster algorithms for rectangular matrix multiplication. In: Proceedings of the 53rd Symposium on Foundations of Computer Science (FOCS), pp. 514–523 (2012)
17. Muthukrshnan, S.: New results and open problems related to non-standard stringology. In: 6th Proceedings of the Combinatorial Pattern Matching (CPM). LNCS, vol. 937, pp. 298–317. Springer (1995)
18. Shapira, A., Yuster, R., Zwick, U.: All-pairs bottleneck paths in vertex weighted graphs. Algorithmica **59**, 621–633 (2011)
19. Vassilevska, V., Williams, R.: Finding a maximum weight triangle in $n^{3-\delta}$ time, with applications. In: Proceedings of the 38th Annual ACM Symposium on Theory of Computing (STOC 2006), pp. 225–231. ACM (2006)
20. Vassilevska, V., Williams, R.: Finding, minimizing, and counting weighted subgraphs. In: Proceedings of the 41st Annual ACM Symposium on Theory of Computing (STOC 2009), pp. 455–463. ACM (2009)
21. Vassilevska, V., Williams, R., Yuster, R.: Finding heaviest H-subgraphs in real weighted graphs, with applications. ACM Trans. Algorithms **6**(3), 44:1–44:23 (2010)
22. Vassilevska Williams, V.: Multiplying matrices faster than Coppersmith–Winograd. In: Proceedings of the 44th Annual ACM Symposium on Theory of Computing (STOC), pp. 887–898 (2012)
23. Yang, B., Chen, J., Lu, E., Zheng, S.Q.: A Comparative study of efficient algorithms for partitioning a sequence into monotone subsequences. In: Proceedings of the 4th Theory and Applications of Models of Computation (TAMC). LNCS, vol. 4484, pp. 46–57. Springer (2007)
24. Zwick, U.: All pairs shortest paths using bridging sets and rectangular matrix multiplication. J. ACM **49**(3), 289–317 (2002)