Available online at www.sciencedirect.com

ScienceDirect

Electronic Notes in Theoretical Computer Science 346 (2019) 557–566

Electronic Notes in Theoretical Computer Science

www.elsevier.com/locate/entcs

The 2D Subarray Polytope

Ivo Koch[a,1] Javier Marenco[b,2]

[a] *Instituto de Industria, Universidad Nacional de General Sarmiento, Argentina*

[b] *Instituto de Ciencias, Universidad Nacional de General Sarmiento, Argentina*

Abstract

Given a d-dimensional array, the *maximum subarray problem* consists in finding an axis-parallel slice of the array maximizing the sum of its entries. In this work we start a polyhedral study of a natural integer programming formulation for this problem when $d = 2$. Such an exploration is motivated by the need of solving large-scale instances of the *rectilinear picture compression problem (RPC)*, a problem which arises in different scenarios. The obtained results can be useful to solve the column generation phase of a branch and price approach for RPC, a technique that applies naturally to this problem. We thus define the *2D subarray polytope*, explore conditions ensuring the validity of linear inequalities, and provide several families of facet-inducing inequalities.

Keywords: maximum subarray problem, integer programming, facets

1 Introduction

The *rectilinear picture compression problem* (RPC) consists in covering all entries with value 1 of a binary matrix $M \in \{0,1\}^{m\times n}$ with a minimum number of submatrices having consecutive rows and consecutive columns (henceforth called *rectangles*), such that each rectangle contains only entries in M with value 1. Although the initial motivation for exploring RPC comes from the compression of monochromatic images (in particular, images coming from rectangular constructs), this problem also has applications in other scenarios.

[1] Email: ikoch@ungs.edu.ar

[2] Email: jmarenco@ungs.edu.ar

https://doi.org/10.1016/j.entcs.2019.08.049

1571-0661/

One such setting, originating challenging instances of RPC, is the synthesis of DNA arrays [7]. As a part of this method, light is selectively allowed through a mask to expose cells in the array, activating the fragments of nucleic acids in that cell. Masks for DNA arrays are built by patterning equipment that generates rectangular regions by a long series of "flashes", each flash producing a rectangle. Since the cost of a mask is proportional to the number of generated rectangles, a cover of a mask with a minimum number of rectangles is desired. This technique is similar to the one used in the semiconductor industry. Semiconductor-based integrated circuit masks usually consist of orthogonal polygons with each side parallel to one of the axes, and these polygons must be covered by the smallest possible number of rectangles [8].

Another application of RPC is in the processing of *access control lists (ACLs)*. ACLs are used in network routers to determine which arriving packets should be forwarded to their destination and which should be dropped. An ACL can be regarded (with some simplification) as a set of tuples (S, D, a), where S and D are source and destination address ranges, respectively, and $a \in \{0, 1\}$ is a forward (1) or deny (0) action. S and D are specified by binary strings of length w or less, where w is the length of an IP address. Minimization of the size of ACLs allows to improve the access control performance of current state-of-the-art routers. The set of rules of an ACL can be modeled as a $2^w \times 2^w$ matrix with an entry for each combination of a source and a destination IP address (columns and rows indexed from 0 to $2^w - 1$). The action of an ACL can be viewed as setting a 1 in an entry if and only if that combination is permitted (i.e., forwarded), and a minimum rectangle cover of the entries with value 1 in the matrix is known as the *ACL minimization problem* [2].

The earliest reference to RPC seems to be due to Masek [10]. In this work, the author showed that RPC is NP-hard; Berman and DasGupta later proved it MaxSNP-hard [4]. The best known polynomial-time approximation guarantee is $O(\sqrt{\log k})$, where k is the number of entries with value 1 in the input matrix [1]. A slightly more general version of the problem has also been studied under a polyhedral approach in [9] and in [11]. In the former, the authors analyze new lower bounds on the optimal cover size based on the fractional solution of the linear programming relaxation of the proposed formulation. The latter discusses two integer programming models for the rectangle cover of a convex polygon.

Given a binary matrix $M \in \{0, 1\}^{m \times n}$, a *rectangle* in M is formally defined to be the entry set $\{(k, \ell) : i_1 \leq k \leq i_2 \text{ and } j_1 \leq \ell \leq j_2\}$ for $1 \leq i_1 \leq i_2 \leq m$ and $1 \leq j_1 \leq j_2 \leq n$. We define $R(M)$ to be the set of rectangles in M that contain only entries with value 1 in M. For each $r \in R(M)$, we introduce a binary variable x_r specifying whether rectangle r is chosen in the cover or not. For every entry (i, j) with $M_{ij} = 1$, at least one rectangle containing (i, j) has to be selected, and we seek to minimize the number of selected rectangles.

$$\min \sum_{r \in R(M)} x_r \tag{1}$$

$$M_{ij} \leq \sum_{r \in R(M):(i,j) \in r} x_r \qquad i = 1, \ldots, m,\ j = 1, \ldots, n \tag{2}$$

$$x_r \in \{0,1\} \qquad\qquad r \in R(M) \tag{3}$$

The number of rectangles in $R(M)$ is polynomial, namely $|R(M)| \in O(n^2m^2)$. However, for a medium- to large-sized input matrix, this formulation quickly leads to an impractically large number of variables. In this setting, a natural solution is a column generation approach, i.e., the dynamic generation of rectangle variables for the linear relaxation of the formulation when strong duality is violated. In this scenario, column generation consists in finding a (weighted) rectangle of negative reduced cost. In other words, we seek a 2-dimensional integer array of maximum weight within M. Efficient resolution of this problem is essential for a branch and price scheme that exploits the described column generation. In addition, the linear programming relaxation of the above formulation turns out to be extremely tight, as shown in the experiments conducted in [9]. This makes the branch and price approach even more promising, since it is more likely that nodes are cut off from the search tree, thus speeding up the search for an optimal solution. Both arguments motivate our study of the maximum 2D subarray polytope, which is the main focus of this work.

2 The maximum subarray problem

Given a d-dimensional real-valued array A, $d \geq 1$, the *maximum subarray problem* consists in finding a contiguous and axis-parallel section of A with maximum sum. We are interested in the case $d = 2$, which corresponds to a 2-dimensional array $A \in \mathbb{R}^{m\times n}$, and asks for row indices $i_1, i_2 \in \{1, \dots, m\}$, $i_1 \leq i_2$, and column indices $j_1, j_2 \in \{1, \dots, n\}$, $j_1 \leq j_2$, such that $\sum_{i=i_1}^{i_2} \sum_{j=j_1}^{j_2} A_{ij}$ is maximum.

We assume $m \leq n$. The classical algorithm described by Bentley solves this problem in $O(m^2n)$ time [3], and subsequent works showed that this problem can be solved in sub-cubic time. Indeed, the algorithm presented in [13] runs in $O(m^2n(\log\log m/\log m)^{1/2}\log(n/m))$ time, and this bound was improved in [12], with an $O(m^2n(\log\log m/\log m)^{1/2})$ worst case.

Although sub-cubic, these algorithms may not be of practical use when m and n are close to real-sized images. Furthermore, in a column generation environment we need not solve the column generation problem to optimality, since a column with a negative reduced cost suffices to continue with the procedure. Hence, fast heuristics may be useful for this problem and, in particular, linear programming-based heuristics (e.g., rounding heuristics) may be of interest. Due to these facts, we are interested in an integer-programming based approach for the maximum subarray problem for $d = 2$.

In this work we start such an issue, by exploring the polytope associated with a natural integer programming formulation of this problem. The final objective of such an undertaking is to identify strong families of valid inequalities that could be useful within a cutting plane procedure for the maximum subarray problem for $d = 2$. Such families of valid inequalities may also be useful within a linear programming-based rounding heuristic, by strengthening the linear relaxation previous to a rounding phase.

The associated polytope may be regarded as a two-dimensional version of the *full interval vectors polytope*, i.e., the convex hull of vectors in $\{0,1\}^n$ having consecutive ones. The corresponding polytope has been studied in [6] and the results therein have inspired some of the results in the current work.

3 The 2D subarray polytope

We consider a real-valued matrix A with m rows and n columns. Denote by $R = \{1,\ldots,m\}$ the set of row indexes, and by $C = \{1,\ldots,n\}$ the set of column indexes. We also define $P = R \times C$ to be the set of entries of A (also called *pixels* in this context). For $(i,j) \in P$, the binary variable x_{ij} takes value 1 if and only if the pixel (i,j) belongs to the represented rectangle.

The *rectangular hull* of a set $S \subseteq P$ of pixels, denoted by $\square(S)$, is the smallest rectangle including all the pixels in S, i.e., $\square(S) = \{(k,\ell) : \min_{(i,j)\in S} i \le k \le \max_{(i,j)\in S} i \text{ and } \min_{(i,j)\in S} j \le \ell \le \max_{(i,j)\in S} j\}$. If $S = \{p, p'\}$ with $p = (i,j)$ and $p' = (i',j')$, then we also denote $\square(S)$ by $\square(p,p')$ and by $\square(i,j,i',j')$. For $(i,j),(i',j') \in P$, we define $\blacksquare(i,j,i',j')$ to be the feasible solution $x \in \{0,1\}^{mn}$ having $x_{k\ell} = 1$ if and only if $(k,\ell) \in \square(i,j,i',j')$.

Definition 3.1 For $m \in \mathbb{Z}_+$ and $n \in \mathbb{Z}_+$, we define $P^{\square}_{m,n} = \operatorname{conv}\big(\{0\} \cup \{\blacksquare(i,j,i',j') : 1 \le i \le i' \le m \text{ and } 1 \le j \le j' \le n\}\big)$.

We now give a formulation for the 2D maximum subarray problem as an optimization problem over $P^{\square}_{m,n}$. For $(i,j) \in P$, A_{ij} is the benefit associated with picking the pixel (i,j). In this setting, the 2D maximum subarray problem can be formulated as follows.

$$\max \sum_{(i,j)\in P} A_{ij} x_{ij}$$

$$x_{ij} + x_{i(j-2)} \le x_{i(j-1)} + 1 \qquad (i,j) \in P,\ j > 2 \tag{4}$$

$$x_{ij} + x_{(i-2)j} \le x_{(i-1)j} + 1 \qquad (i,j) \in P,\ i > 2 \tag{5}$$

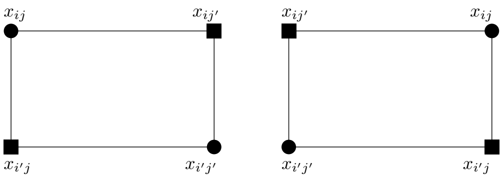
$$x_{ij} + x_{i'j'} \le \frac{x_{ij'} + x_{i'j}}{2} + 1 \qquad (i,j),(i'j') \in P,\ i < i',\ j \ne j' \tag{6}$$

$$x_{ij} \in \{0,1\} \qquad (i,j) \in P \tag{7}$$

Constraints (4) (resp. (5)) force the variables in the same row (resp. column) to be contiguous. Constraints (6) assure that if pixels (i,j) and (i',j'), with $i < i'$, are part of the selected rectangle, then pixels (i,j') and (i',j) must be contained in the rectangle as well. This family of constraints is illustrated in Fig. 1.

The convex hull of feasible solutions to (4)-(7) coincides with $P^{\square}_{m,n}$. Note that we allow the empty solution to be a feasible solution, namely $0 \in P^{\square}_{m,n}$. This implies the following result.

Proposition 3.2 *$P^{\square}_{m,n}$ is full-dimensional.*



(a) Case where $j' > j$ (b) Case where $j' < j$

Fig. 1. Constraints (6) of the formulation for the 2D maximum subarray problem. The rectangles represent $\square(i,j,i',j')$, and the square dots represent the x-variables that must be set to 1 if $x_{ij} = x_{i'j'} = 1$.

4 Facets of the 2D subarray polytope

Let $\pi \in \mathbb{Z}^{mn}$. For $c \in \mathbb{Z}$, we define $I_c^\pi = \{(i,j) \in P : \pi_{ij} = c\}$ to be the set of pixels with coefficient c in π. We also define $I_{>0}^\pi$ to be the set of pixels with positive coefficient in π, and $I_{<0}^\pi$ to be the set of pixels with negative coefficient in π. These definitions allow us to provide a general characterization of valid inequalities for $P_{m,n}^\square$.

Theorem 4.1 *Let $\pi \in \mathbb{Z}^{mn}$ and $\pi_0 \in \mathbb{Z}$. The inequality $\pi x \leq \pi_0$ is valid for $P_{m,n}^\square$ if and only if*

$$\sum_{(i,j)\in S} \pi_{ij} + \sum_{(i,j)\in\square(S)\cap I_{<0}^\pi} \pi_{ij} \;\leq\; \pi_0$$

for every $S \subseteq I_{>0}^\pi$.

Although difficult to check in practice, the condition ensuring validity in Theorem 4.1 will be useful in the next sections.

4.1 Facet-inducing inequalities with coefficients in $\{-1,0,1\}$

We first explore valid inequalities with coefficients in $\{-1,0,1\}$. In this case, Theorem 4.1 implies the following characterization for validity.

Corollary 4.2 *Let $\pi \in \{-1,0,1\}^{mn}$. The inequality $\pi x \leq 1$ is valid for $P_{m,n}^\square$ if and only if $|\square(S) \cap I_{-1}^\pi| \geq |S| - 1$ for every $S \subseteq I_1^\pi$.*

We say that a pixel $(k,\ell) \in P$ is *reachable in* π from the pixel $(i,j) \in P$ if $\square(i,j,k,\ell)\backslash\{(i,j)\} \subseteq I_0^\pi$, i.e., all pixels in the rectangular hull $\square(i,j,k,\ell)$ have coefficient 0 in π, with the exception of (i,j).

Theorem 4.3 *Let $\pi x \leq 1$ be a valid inequality with $\pi \in \{-1,0,1\}^{mn}$. If (a) every pixel in I_0^π is reachable from some pixel in I_1^π and (b) for every $p \in I_{-1}^\pi$ there exist $q,q' \in I_1^\pi$ such that $\square(q,q') \cap I_1^\pi = \{q,q'\}$ and $\square(q,q') \cap I_{-1}^\pi = \{p\}$, then $\pi x \leq 1$ defines a facet of $P_{m,n}^\square$.*

Theorem 4.3 allows us to derive several families of facet-inducing inequalities $\pi x \leq 1$ for $P_{m,n}^\square$ with coefficients in $\{-1,0,1\}$ (see Fig. 2 for an example), and is the starting point for the subsequent theorems. It is important to note that Theorem 4.3 does not characterize all facet-inducing inequalities with coefficients

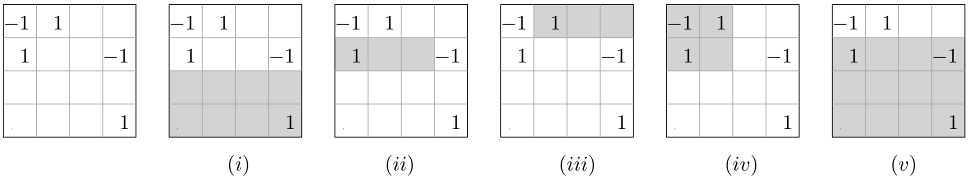


Fig. 2. A valid inequality $\pi x \leq 1$ that verifies the hypotheses of Theorem 4.3. The inequality is represented by specifying the nonzero coefficients of π in the corresponding pixels of the input matrix. The gray rectangles in subfigures $(i) - (iii)$ show that Condition (a) of Theorem 4.3 is verified, while the gray rectangles in (iv) and (v) show that Condition (b) is also met.

in $\{-1, 0, 1\}$. Indeed, some of the following facet-inducing inequalities do not stem from this result directly.

Theorem 4.4 *Let* $\pi \in \{-1, 0, 1\}^{mn}$*. If* $I_1^\pi = \{(i_1, j_1), (i_2, j_2)\}$ *and* $I_{-1}^\pi = \{(k, \ell)\} \subseteq \square(i_1, j_1, i_2, j_2)$*, then* $\pi x \leq 1$ *is valid and facet-inducing for* $P_{m,n}^\square$*.*

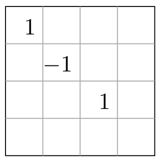


Fig. 3. A valid inequality $\pi x \leq 1$ that verifies the hypotheses of Theorem 4.4, with $(i_1, j_1) = (1, 1)$, $(i_2, j_2) = (3, 3)$, and $(k, l) = (2, 2)$.

Theorem 4.4 is a corollary of Theorem 4.3 when $i_1 < k < i_2$ and $j_1 < \ell < j_2$ (see Fig. 5 for an example). Facetness in the cases $k \in \{i_1, i_2\}$ and $\ell \in \{j_1, j_2\}$ requires a slightly different proof.

Theorem 4.5 *Let* $\pi \in \{-1, 0, 1\}^{mn}$ *with* $|I_1^\pi| = |I_{-1}^\pi| + 1$*. If* $I_1^\pi = \{(i_t, j_t)\}_{t=1}^k$ *with* $i_t \leq i_{t+1}$ *and* $j_t \leq j_{t+1}$ *for* $t = 1, \ldots, k-1$*, and* $\square(i_t, j_t, i_{t+1}, j_{t+1})$ *contains exactly one pixel from* I_{-1}^π *for* $t = 1, \ldots, k-1$*, then* $\pi x \leq 1$ *is valid and facet-inducing for* $P_{m,n}^\square$*.*

The family of facet-inducing inequalities specified by Theorem 4.5 includes the *interval constraints* $x_{i_1,j} - x_{i_2,j} + x_{i_3,j} - x_{i_4,j} + \cdots + x_{i_{2k+1},j} \leq 1$, with $i_t < i_{t+1}$ for $t = 1, \ldots, 2k$, coming from the full interval vectors polytope [6].

Theorem 4.6 *Let* $\pi \in \{-1, 0, 1\}^{mn}$*. If* $I_1^\pi = \{p_1, p_2, p_3\}$ *and* $I_{-1}^\pi = \{q_1, q_2\}$ *such that* $I_{-1}^\pi \cap \square(p_1, p_2) \cap \square(p_1, p_3) = \{q_1\}$ *and* $I_{-1}^\pi \cap [\square(p_2, p_3) \backslash \square(p_1, p_2)] = \{q_2\}$*, then* $\pi x \leq 1$ *is valid and facet-inducing for* $P_{m,n}^\square$*.*

Theorem 4.7 *Let* $\pi \in \{-1, 0, 1\}^{mn}$*. If* $I_1^\pi = \{p_1, p_2, p_3\}$ *and* $I_{-1}^\pi = \{q_1, q_2, q_3\}$ *such that*

- $I_{-1}^\pi \cap [\square(p_1, p_2) \backslash (\square(p_2, p_3) \cup \square(p_1, p_3))] = \{q_1\}$,
- $I_{-1}^\pi \cap [\square(p_2, p_3) \backslash (\square(p_1, p_2) \cup \square(p_1, p_3))] = \{q_2\}$,
- $I_{-1}^\pi \cap [\square(p_1, p_3) \backslash (\square(p_1, p_2) \cup \square(p_2, p_3))] = \{q_3\}$,

then $\pi x \leq 1$ *is valid and facet-inducing for* $P_{m,n}^\square$*.*

1				
	−1			
	1			
			−1	
				1

Fig. 4. A valid inequality $\pi x \leq 1$ that verifies the hypotheses of Theorem 4.6, with $p_1 = (1, 1)$, $p_2 = (3, 2)$, $p_3 = (5, 5)$, and $q_1 = (2, 2)$, $q_2 = (4, 4)$.

−1		1		
			−1	
				1
			−1	
1				

Fig. 5. A valid inequality $\pi x \leq 1$ that verifies the hypotheses of Theorem 4.7, with $p_1 = (1, 3)$, $p_2 = (3, 5)$, $p_3 = (5, 1)$, and $q_1 = (2, 4)$, $q_2 = (4, 4)$, $q_3 = (1, 1)$.

It is interesting to note that Theorems 4.4, 4.5 and 4.6 provide facet-inducing inequalities similar to the interval constraints of the full interval vectors polytope, namely inequalities with $|I_1^\pi| = |I_{-1}^\pi| + 1$. Theorem 4.7, on the other hand, provides a facet-inducing inequality with $|I_1^\pi| = |I_{-1}^\pi|$. It would be interesting to explore whether this construction can be extended to $|I_1^\pi| > 3$.

Using the same proof technique as in Theorem 4.3, we can show the following generalization.

Theorem 4.8 *Let $\pi \in \{-1, 0, 1\}^{mn}$ with $|I_1^\pi| = |I_{-1}^\pi| + 1$. Assume (a) every pixel in I_0^π is reachable from some pixel in I_1^π. If there exists a list $L = \{R_1, R_2, \ldots, R_k\}$ of subrectangles such that (b) every pixel in I_{-1}^π is contained in R_i for some $i \in \{1, \ldots, k\}$, (c) $|R_i \cap I_1^\pi| = |R_i \cap I_{-1}^\pi| + 1$ for $i = 1, \ldots, k$, and (d) $|I_{-1}^\pi \cap (R_i \setminus \bigcup_{j=1}^{i-1} R_j)| = 1$ for $i = 1, \ldots, k$, then $\pi x \leq 1$ is facet-inducing for $P_{m,n}^\square$.*

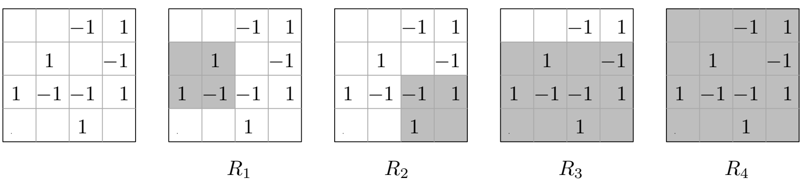


Fig. 6. A valid inequality $\pi x \leq 1$ that verifies the hypotheses of Theorem 4.8. The gray rectangles in the subfigures represent the rectangles R_i of the list L of the hypothesis. Notice that hypothesis (b) of Theorem 4.3 is not satisfied.

Interestingly, hypotheses (b)-(d) of Theorem 4.8 seem to be necessary for facetness of inequalities $\pi x \leq 1$ with $\pi \in \{-1, 0, 1\}^{mn}$ having $|I_1^\pi| = |I_{-1}^\pi| + 1$ and satisfying the hypothesis (a). We performed an exhaustive computational verification of all facet-inducing inequalities of $P_{3,5}^\square$, $P_{4,4}^\square$ and $P_{3,6}^\square$, with the help of the PORTA [5] software package. In all cases (approx. 7700 facet-inducing inequalities for $P_{3,5}^\square$, 9900 facet-inducing inequalities for $P_{4,4}^\square$, and 47500 facet-inducing inequalities for $P_{3,6}^\square$) a list L satisfying the hypotheses was indeed found. Due to impractical running times, larger instances could not be checked.

4.2 Facet-inducing inequalities with coefficients in $\{-2, 0, 1\}$ and $\{-3, 0, 1\}$

We now explore valid inequalities with at least one coefficient greater than 1 in absolute value.

Theorem 4.9 *Let $\pi \in \{-2, 0, 1\}^{mn}$. If $I_1^\pi = \{(i_1, j_1), (i_2, j_2), (i_3, j_3)\}$ with $i_1 < i_2$, $j_1 < j_2$, $i_3 \leq \min\{i_1, i_2\}$, $j_3 \geq \max\{j_1, j_2\}$, and $I_{-2}^\pi = \{(i_1, j_2)\}$, then $\pi x \leq 1$ is valid and facet-inducing for $P_{m,n}^\square$.*

1		−2	1
		1	

Fig. 7. A valid inequality $\pi x \leq 1$ that verifies the hypotheses of Theorem 4.9, with $(i_1, j_1) = (1, 1)$, $(i_2, j_2) = (2, 3)$, and $(i_3, j_3) = (1, 4)$.

Note that the hypotheses concerning the pixels in I_1^π and in I_{-2}^π imply that $(i_1, j_2) \in \square(S)$ for any $S \subseteq I_1^\pi$ with $|S| \geq 2$, so Theorem 4.1 ensures that $\pi x \leq 1$ is indeed valid (see Fig. 7 for an example). Facetness is implied by an argument generalizing the proof of Theorem 4.3.

Theorem 4.10 *Let $\pi \in \{-3, 0, 1\}^{mn}$. If $I_1^\pi = \{p_1, p_2, p_3, p_4\}$ with $\square(p_1, p_2) \cap \square(p_3, p_4) = \{q\} = I_{-3}^\pi$, then $\pi x \leq 1$ is valid and facet-inducing for $P_{m,n}^\square$.*

		1	
1		−3	1
		1	

Fig. 8. A valid inequality $\pi x \leq 1$ that verifies the hypotheses of Theorem 4.10, with $p_1 = (3, 1)$, $p_2 = (2, 3)$, $p_3 = (3, 4)$, $p_4 = (4, 3)$, and $q = (3, 3)$.

It would be interesting to search for further configurations originating valid and facet-inducing inequalities with such large coefficients. In our experiments with small instances, we could not find facet-inducing inequalities with (normalized) integer coefficients outside the range $\{-3, \ldots, 4\}$.

5 Conclusions

We have started in this work a polyhedral study of the polytope associated with a natural integer programming formulation for the maximum subarray problem for $d = 2$. Our objective is to identify strong families of valid inequalities that could be useful within a cutting plane procedure, or within a linear programming-based rounding heuristic for this problem. The final goal of this analysis is to obtain a strong column generation algorithm for RPC.

We have found several families of facet-inducing inequalities, many of them with coefficients in $\{-1, 0, 1\}$. From a polyhedral point of view, it would be desirable to achieve a more thorough theoretical treatment of these inequalities as, e.g., providing necessary and sufficient conditions ensuring facetness for valid inequalities with coefficients in $\{-1, 0, 1\}$. We believe that Theorem 4.8 yields a promising basis for such a characterization, and proof of the remaining necessary condition could be addressed in a future work. Further, since polynomial algorithms for the maximum subarray problem are known, we pose the existence of an integer programming model for the problem with a totally unimodular matrix as an open question.

From a practical point of view, the computational complexity of the separation problems associated with the introduced families is of interest, in particular since exhaustive enumerations do not provide polynomial-time algorithms for all of these problems and, furthermore, may not be practical in medium- to large-sized instances. The design of fast heuristics for separating these families could be of practical interest as well.

References

[1] Anil Kumar, V. S. and H. Ramesh, *Covering rectilinear polygons with axis-parallel rectangles*, in: *Proceedings of the Thirty-first Annual ACM Symposium on Theory of Computing*, STOC '99 (1999), pp. 445–454.
URL http://doi.acm.org/10.1145/301250.301369

[2] Applegate, D. A., G. Calinescu, D. S. Johnson, H. Karloff, K. Ligett and J. Wang, *Compressing rectilinear pictures and minimizing access control lists*, in: *Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '07 (2007), pp. 1066–1075.
URL http://dl.acm.org/citation.cfm?id=1283383.1283498

[3] Bentley, J., *Programming Pearls - Perspective on Performance*, Comm. ACM **27** (1984), pp. 1087 – 1092.

[4] Berman, P. and B. DasGupta, *Complexities of efficient solutions of rectilinear polygon cover problems*, Algorithmica **17** (1997), pp. 331–356.
URL https://doi.org/10.1007/BF02523677

[5] Christof, T., *PORTA - a Polyhedron Representation Transformation Algorithm*, version 1.4.1.
URL https://wwwproxy.iwr.uni-heidelberg.de/groups/comopt/software/PORTA/

[6] Dahl, G., *Polytopes related to interval vectors and incidence matrices*, Linear Algebra and its Applications **435** (2011), pp. 2955 – 2960.
URL http://www.sciencedirect.com/science/article/pii/S0024379511004319

[7] Hannenhalli, S., E. Hubbell, R. Lipshutz and P. A. Pevzner, "Advances in Biochemical Engineering/Biotechnology," Chip Technology **77**, Springer, Berlin, Heidelberg, 2002 .

[8] Hegeds, A., *Algorithms for covering polygons by rectangles*, Computer-Aided Design **14** (1982), pp. 257 – 260.
URL http://www.sciencedirect.com/science/article/pii/001044858290104X

[9] Heinrich-Litan, L. and M. E. Lübbecke, *Rectangle covers revisited computationally*, J. Exp. Algorithmics **11** (2007).
URL http://doi.acm.org/10.1145/1187436.1216583

[10] Masek, W. J., *Some NP-Complete Set Covering Problems* (1979), MIT, Unpublished Manuscript.

[11] Scheithauer, G., Y. Stoyan and T. Romanova, *Integer linear programming models for the problem of covering a polygonal region by rectangles*, Radioelectronics & Informatics **2** (2009), pp. 4–13. URL https://cyberleninka.ru/article/n/integer-linear-programming-models-for-the-problem-of-covering-a-polygonal-region-by-rectangles

[12] Takaoka, T., *Efficient algorithms for the maximum subarray problem by distance matrix multiplication*, Electronic Notes in Theoretical Computer Science **61** (2002), pp. 191 – 200, CATS '02, Computing: the Australasian Theory Symposium.
URL http://www.sciencedirect.com/science/article/pii/S1571066104003135

[13] Tamaki, H. and T. Tokuyama, *Algorithms for the maximum subarray problem based on matrix multiplication*, in: *Proceedings of the Ninth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '98 (1998), pp. 446–452.
URL http://dl.acm.org/citation.cfm?id=314613.314823