ALGORITHMIC COMPUTATION OF THE UNIVERSAL
GRÖBNER BASIS OF TORIC IDEALS OF GRAPHS

YANNIS C. STAMATIOU AND CHRISTOS TATAKIS

Abstract. Let $G$ be an undirected graph and $I_G$ be its corresponding toric ideal. In this paper we give an algorithm that outputs its Universal Gröbner basis based on a recent, efficiently computable algorithmic characterization of its the Graver basis.

1. Introduction

A Gröbner basis is a specific generating set of an ideal over a polynomial ring. It has extremely useful algebraic properties and it is relatively easy to extract information about the ideal, given its Gröbner basis. The study of the Gröbner basis set has become a major research topic in commutative algebra, combinatorics and computer science. Gröbner basis theory provides the foundations for many algorithms in commutative algebra and algebraic geometry. Its importance stems from its wide applicability in problems coming from diverse disciplines such as mathematics, combinatorial and computer design theory, symbolic summation, integer programming, engineering, computer technology and cryptography.

The concept of a Gröbner basis was introduced by Buchberger at 1965 who named this set after his supervisor W. Gröbner. Buchberger’s algorithm is the most well known algorithmic method for computing a Gröbner basis for an ideal $I$ of a polynomial ring. Most of the symbolic computation software packages, such as CoCoA, Macaylay, Mathematica, Maple and Singular, include algorithms for computing this set. For more information on Gröbner bases and corresponding algorithms, see [1],[2],[3].

Even though the monomial ordering is fixed, the corresponding Gröbner bases are not unique. The polynomial ring $\mathbb{K}[x_1, \ldots, x_n]$ for $n \geq 2$ has infinitely many distinct monomial orders and, therefore, it has infinitely Gröbner bases. Nevertheless, for a fixed nonzero ideal $I$ there exist finitely many different reduced Gröbner bases for $I$. The universal Gröbner basis is a finite subset of $I$ and it is a Gröbner basis for the ideal with respect to all admissible term orders (see [8]). It is the union of all reduced Gröbner bases $G_<$ of the ideal $I$, as $<$ runs over all term orders. Universal Gröbner bases exist for every ideal in $\mathbb{K}[x_1, \ldots, x_n]$. They were introduced by V. Weispfenning [12] and N. Schwartz [7].

In [8], B. Sturmfels gave algorithms for computing a Graver basis by Lawrence lifting of $A$ and Universal Gröbner bases for toric ideals (with an algebraic geometry view). Moreover Sturmfels gave an algorithm, which takes as input this set, in order to produce the State polytope of a toric ideal $I$. There are not many classes of ideals for which we know their Universal Gröbner basis. In general, characterizing...
and computing the Universal Gröbner basis is a very difficult and computationally demanding problem.

One of the known classes, are the toric ideals of graphs. The structure of the Universal Gröbner basis of a toric ideal of a graph $G$, was characterized theoretically by Ch. Tatakis and A. Thoma, see [9, Theorem 3.4]. The goal of this paper is to transform this Theorem into an algorithm for computing this set.

Let $A = \{a_1, \ldots, a_m\} \subseteq \mathbb{N}^n$ be a vector configuration in $\mathbb{Q}^n$ and $NA := \{l_1a_1 + \cdots + l_ma_m \mid l_i \in \mathbb{N}\}$ the corresponding affine semigroup. We grade the polynomial ring $\mathbb{K}[x_1, \ldots, x_m]$ over an arbitrary field $\mathbb{K}$ by the semigroup $NA$ setting $\deg_A(x_i) = a_i$ for $i = 1, \ldots, m$. For $u = (u_1, \ldots, u_m) \in \mathbb{N}^m$, we define the $A$-degree of the monomial $x^u := x_1^{u_1} \cdots x_m^{u_m}$ to be

$$\deg_A(x^u) := u_1a_1 + \cdots + u_m a_m \in \mathbb{N}A.$$  

The toric ideal $I_A$ associated to $A$ is the prime ideal generated by all the binomials $x^u - x^v$ such that $\deg_A(x^u) = \deg_A(x^v)$, see [8]. For such binomials, we set $\deg_A(x^u - x^v) := \deg_A(x^u)$. An irreducible binomial $x^u - x^v$ in $I_A$ is called primitive if there exists no other binomial $x^w - x^z$ in $I_A$ such that $x^w$ divides $x^u$ and $x^z$ divides $x^v$. The set of primitive binomials forms the Graver basis of $I_A$ and is denoted by $Gr_A$.

The relation between the Graver basis and the universal Gröbner basis, which is denoted by $\mathcal{U}_A$, for a toric ideal $I_A$ was described by B. Sturmfels [3]:

**Proposition 1.1.** For any toric ideal $I_A$ we have $\mathcal{U}_A \subset Gr_A$.

We note that the elements of the universal Gröbner basis belong to the Graver basis. Therefore the knowledge of the Graver basis for a toric ideal of any graph plays a key role for computing the $\mathcal{U}_A$. In the next section we analyze the set $Gr_A$, from which we compute the $\mathcal{U}_A$ of a toric ideal of any graph $G$.

2. An introduction to toric ideals of graphs and its Graver basis

In this section we first give some basic elements of graph theory which will be useful in the description of the Graver and Universal Gröbner basis of a toric ideal of a graph $G$.

Let $G$ be a finite undirected connected graph with vertices $V(G) = \{v_1, \ldots, v_n\}$ and edges $E(G) = \{e_1, \ldots, e_m\}$. Let $\mathbb{K}[e_1, \ldots, e_m]$ be the polynomial ring in the $m$ variables $e_1, \ldots, e_m$ over a field $\mathbb{K}$. We will associate each edge $e = (v_i, v_j) \in E(G)$ with the element $a_e = v_i + v_j$ in the free abelian group $\mathbb{Z}^n$ with the canonical basis the set of vertices of $G$. The toric ideal of a graph $G$ is denoted by $I_G$ and is defined as the toric ideal $I_{AG}$ in $\mathbb{K}[e_1, \ldots, e_m]$, where $AG = \{a_e \mid e \in E(G)\} \subset \mathbb{Z}^n$.

A walk connecting $v_{i_1} \in V(G)$ and $v_{i_{s+1}} \in V(G)$ is a finite sequence of the form

$$w = (\{v_{i_1}, v_{i_2}\}, \{v_{i_2}, v_{i_3}\}, \ldots, \{v_{i_s}, v_{i_{s+1}}\})$$

with each $e_{ij} = \{v_i, v_j\} \in E(G)$. The length of the walk $w$ is the number $s$ of edges of the walk. An even (respectively odd) walk is a walk of even (respectively odd) length. A walk $w = (\{v_1, v_2\}, \{v_2, v_3\}, \ldots, \{v_{s}, v_{i_{s+1}}\})$ is called closed if $v_{i_{s+1}} = v_1$. A cycle is a closed walk

$$\{v_1, v_2, v_3, \ldots, v_k\}$$

with $v_k \neq v_1$, for every $1 \leq k < j \leq s$. Note that, although the graph $G$ has no multiple edges, the same edge $e$ may appear more than once in a walk. In this case
e is called a multiple edge of the walk w. Given an even closed walk of the graph G, \( w = (e_{i_1}, e_{i_2}, \ldots, e_{i_2q}) \) we denote by
\[
E^+(w) = \prod_{k=1}^{q} e_{i_{2k-1}}, \quad E^-(w) = \prod_{k=1}^{q} e_{i_{2k}}
\]
and by \( B_w \) the binomial
\[
B_w = \prod_{k=1}^{q} e_{i_{2k-1}} - \prod_{k=1}^{q} e_{i_{2k}}.
\]
We remark that \( B_w \in I_G \). Moreover, it is known that the toric ideal \( I_G \) is generated by binomials of this form, see [11].

We denote by \( \mathbf{w} \) the subgraph of \( G \) with vertices the vertices of the walk and edges the edges of the walk \( w \). If \( W \) is a subset of the vertex set \( V(G) \) of \( G \) then the induced subgraph of \( G \) on \( W \) is the subgraph of \( G \) whose vertex set is \( W \) and whose edge set is \( \{ \{v, u\} \in E(G) | v, u \in W \} \). The walk \( w \) is primitive if and only if the binomial \( B_w \) is primitive. A cut edge (respectively cut vertex) is an edge (respectively vertex) of the graph whose removal increases the number of connected components of the remaining subgraph. A block is a maximal connected subgraph of a given graph \( G \) which does not contain a cut vertex.

A necessary and sufficient characterization of the primitive walks of an undirected graph, i.e. of the Graver basis of the corresponding toric ideal \( I_G \), was given by E. Reyes, Ch. Tatakis and A. Thoma in [6, Theorem 3.2]. The next Corollary was given by the same authors and describes the structure of the underlying graph of a primitive walk, i.e. the elements \( B_w \in I_G \) that belong to the Graver basis.

**Corollary 2.1.** Let \( G \) be a graph and \( W \) a connected subgraph of \( G \). The subgraph \( W \) is the graph \( \mathbf{w} \) of a primitive walk \( w \) if and only if

1. \( W \) is an even cycle or
2. \( W \) is not biconnected and
   a. every block of \( W \) is a cycle or a cut edge and
   b. every cut vertex of \( W \) belongs to exactly two blocks and separates the graph in two parts, the total number of edges of the cyclic blocks in each part is odd.

3. **Algorithmic description of the Universal Gröbner Basis of \( I_G \)**

In this section we state the main result of this paper which gives an algorithmic description of the Universal Gröbner Basis (UGB) of a toric ideal of a graph \( G \).

In [5] M. Ogawa, H. Hara and A. Takemura gave an algorithm for sampling elements from the Graver basis set of \( I_G \), which was associated with a simple undirected graph for testing the beta model of graphs by Markov chains, based on Monte Carlo methods. In general, the Graver basis of graphs can be computed by symbolic computation software packages such as the 4ti2 (see [10]).

For the algorithm that we will present, its input is the set of the primitive elements of a toric ideal of a graph \( G \). In [4] Ch. Tatakis and A. Thoma gave a necessary and sufficient characterization of the Universal Gröbner Basis of \( I_G \). We transform this characterization into an algorithm for computing the Universal Gröbner Basis of any graph. We, first, give the basic definitions and the theorem that characterizes the structure of the UGB (see [9] for further details).
Every even primitive walk \( w = (e_i, \ldots, e_{i_{2k}}) \) partitions the set of edges in the two sets \( w^+ = \{ e_i | j \text{ odd} \} \), \( w^- = \{ e_i | j \text{ even} \} \), otherwise the binomial \( B_w \) would not be irreducible. The edges of \( w^+ \) are called odd edges of the walk and those of \( w^- \) even. Note that for an even closed walk whether an edge is even or odd depends on the edge that we start counting from. Thus, it is not important to identify whether an edge is even or odd but to separate the edges into two disjoint classes.

**Definition 3.1.** [9] A cyclic block \( B_i \) of a primitive walk \( w \) is called pure if all edges of \( B_i \) are either in \( w^+ \) or in \( w^- \). A primitive walk \( w \) is called mixed if no cyclic block of \( w \) is pure.

The following Theorem describes the elements of the universal Gröbner basis of \( I_G \), for any undirected graph \( G \).

**Theorem 3.2.** [9] Let \( w \) be a primitive walk. \( B_w \) belongs to the universal Gröbner basis of \( I_G \) if and only if \( w \) is mixed.

From this theorem, the next corollary follows.

**Corollary 3.3.** Let \( w \) be a primitive walk of a graph \( G \). Then for the binomial \( B_w \) it holds \( B_w \in U_A \) if and only if the set of its odd edges and the set of its even edges do not contain a cycle of \( w \).

**Proof.** Let \( w \) be a primitive walk of a graph \( G \). From Theorem 3.2, \( B_w \in U_A \) if and only if \( w \) is mixed. By definition, \( w \) is mixed if and only if all of its blocks are not pure, i.e. for every block \( B_i \) of its blocks, \( E(B_i) \not\subseteq E(w^+) \) and \( E(B_i) \not\subseteq E(w^-) \). Since \( w \) is primitive, from Theorem 2.1 it follows that all of its blocks are cycles (due to the previous theorem, we are interested only for the cyclic blocks of the walk \( w \)). The result now follows. \( \square \)

Before we give the algorithm, we will describe an example.

**Example 3.4.** Let \( w = \{ e_1, e_2, e_3, e_5, e_6, e_7, e_8, e_9, e_{10}, e_{11}, e_{12} \} \) be an even walk which we also see in the next figure. Let

\[
B_w = e_1e_3 e_5 e_7e_9 e_{11} - e_2 e_4 e_6 e_8 e_{10} e_{12}
\]

be its corresponding binomial.

The above graph \( W \) is not biconnected, each of its blocks is a cycle and every cut vertex of \( W \) belongs to exactly two blocks and separates the graph in two
parts, and the total number of edges of the cyclic blocks in each part is odd. From Theorem 2.1 the walk \( w \) is primitive and for the corresponding binomial \( B_w \), it holds that \( B_w \in Gr_A \). Also \( w \) is not mixed because of the existence of the block \( B_1 = \{e_4, e_8, e_{12}\} \) which is not pure, since all of its edges \( \{e_4, e_8, e_{12}\} \) belong to \( w^- \). Therefore from Theorem 2.2 the binomial \( B_w \notin U_A \). We can also see that in the set of the cycles of \( w \), there is a cycle \( c = (e_4, e_8, e_{12}) \) by application of the previous Corollary.

Now we are ready to describe our algorithm, which computes the Universal Gröbner basis for a toric ideal of a graph \( G \). Its correctness is guaranteed by Corollary 3.3. In [9, Corollary 4.2.] Ch. Tatakis and A. Thoma found an upper bound for the degrees of the primitive elements of \( I_G \). This bound is described in the next proposition. It will be useful in estimating the computational complexity of the algorithm.

**Proposition 3.5.** [9] Let \( G \) be a graph with \( n \) vertices, \( n \geq 4 \). The largest degree \( d \) of any binomial in the Graver basis (and in the universal Gröbner basis) for \( I_G \) is \( d \leq n - 2 \).

The algorithm takes as input the elements of the Graver basis of a graph. At each iteration, the algorithm considers one by one the elements \( B_w \in Gr_A \). For each such element, it repeatedly (loop at line 5) looks for a vertex of degree 2 at line 6. If no such vertex exists, all cycles have been eliminated, the test in line 27 was negative for all of them, and we are left with cut edges alone. In this case, the algorithm is directed to line 33, accepting \( B_w \) as an element of the universal Gröbner basis \( U_A \) and takes the next one at line 3.

If a vertex of degree 2 exists, then the algorithm attempts to detect a cycle (if one exists) following consecutive edges (lines 7 through 15). During this process, either an already visited vertex will appear for the first time or a vertex of degree 1 that belongs to a cut edge (only these two cases may arise due to the structure of the elements of the Graver basis and they are checked in line 15).

In the first case, a cycle has been detected. Then the vertices of the cycle are followed, storing the visited edges in order to form the cycle (lines 21 through 26). This set of edges, stored in the array \( T \), that form the cycle are checked whether it is a subset of the edges in the positive part of \( B_w \) or the negative one (line 27). If one of these cases is true, then \( B_w \) is not an element of the Universal Gröbner basis. If this is not the case, the algorithm deletes the edges of the cycle (line 31) and starts the cycle detection process again.

In the second case, i.e. a vertex of degree 1 is reached (i.e. the check in line 16 succeeds) belonging to a cut edge, the algorithm reverses its way (lines 17 through 20), looking for a vertex of degree 3, in which case it has hit a cycle or it a vertex of degree 1 again, in which the algorithm had followed a path of cut edges (both these cases are checked in line 20). In both cases, the algorithm deletes the cut edges (line 31) and repeats the cycle detection process in line 31.

The formal statement of this process, in pseudocode, follows below.

**ALGORITHM UGB**

**INPUT:**

The elements \( B_w \in Gr_A \), each split into its two parts, i.e. \( B_w^+ \) and \( B_w^- \), as subsets of \( E(G) \) and \( A_{G,w} \), which is a description of \( w \) as a subgraph of \( G \).
OUTPUT: the elements of the universal Gröbner basis $U_A$.

LOCAL VARIABLES:
- $T$: a $1 \times 2(n - 2)$ array of integers (vertex indices) that is used in the cycle finding process.
- stop: a boolean variable.
- $j, k, i, l, r$: integers.
- $U$: a set of graph edges (i.e. non-ordered pairs of integers).

1. begin algorithm
2. $U_A \leftarrow \emptyset$
3. for all elements $B_w \in Gr_A$
4. stop $\leftarrow$ false
5. while stop $=$ false
6. if $\exists i$ such that $\deg(v_i) = 2$, set $i$ to such a value
7. Initialize $T$ to contain 0 in all positions.
8. $r \leftarrow 1$
9. $T[r] \leftarrow i$
10. $U \leftarrow \emptyset$
11. repeat
12. Select $j$, such that $j \neq T[r] \text{ and } \{v_{T[r]}, v_j\} \in B_w$
13. $r \leftarrow r + 1$
14. $T[r] \leftarrow j$
15. until $j$ is contained in the array $T$ or $\deg(v_j) = 1$
16. if $\deg(v_j) = 1$ // We ended at a cut edge.
17. repeat
18. $U \leftarrow U \cup \{v_{T[r - 1]}, v_{T[r]}\}$
19. $r \leftarrow r - 1$
20. until $\deg(v_{T[r]}) = 3$ or $r = 1$
21. else-if line 16
22. Let $l$ be the first position in $T$ where $j$ is stored
23. repeat
24. $U \leftarrow U \cup \{v_{T[r - 1]}, v_{T[r]}\}$
25. $r \leftarrow r - 1$
26. until $r = l$
27. if $U \subseteq B^+_w$ or $U \subseteq B^-_w$
28. stop $\leftarrow$ true
29. end if-line 27
30. end if-line 16
31. $E \leftarrow E - U$ // updating, also, the degrees of the vertices.
32. else-if line 6 // We are, now, left with the cut edges alone, i.e. $B_w \in U_A$.
33. stop $\leftarrow$ true
34. $U_A \leftarrow U_A \cup B_w$
35. end if-line 6
36. end while-line 5
37. end for-line 3
38. end algorithm

In the next Theorem we determine the computational complexity of the algorithm UGB and show that in order to decide algorithmically whether an single element of the Graver basis of $I_G$ belongs also in the $U_A$ it requires polynomial computational steps.

**Theorem 3.6.** The Algorithm UGB correctly computes the set $U_A$ for a toric ideal of a given graph $G$ and its time (i.e. algorithm steps) complexity is $|Gr_A| \cdot \max_w |B_w|^3$. 
Proof. As we discussed before, Corollary \[3.3\] guarantees the correctness of the algorithm.

With respect to the complexity analysis, the algorithm performs \(|Gr_A|\) iterations (lines 3 to 37) in order to check all the elements \(B_w \in Gr_A\). For each element \(B_w\) the algorithm performs iterations in lines 5 to 36 in order to isolate all its cycles and test whether the condition in line 27 holds, in which case \(B_w \not\in U_A\). Otherwise, if for all detected cycles (i.e. when only cut edges remain - see condition in line 6) the condition in line 27 does not hold, then \(B_w \in U_A\) and \(B_w\) is included in the set \(U_A\) in line 34 and the algorithm stops. This loop, thus, runs as long as a vertex of degree other than 1 exists in graph of \(B_w\). Since at each step a cycle is eliminated (if it exists), we conclude that, in the worst case, \(|B_w|\) iterations will be needed to discard edges of degree other than 1.

Within the loop from lines 5 to 36, there are two mutually exclusive (due to the condition in line 6) fragments: the one in lines 7 to 31 and the other in lines 33 to 34.

The first fragment, from line 7 to 31, has one iterative structure in lines 11 to 15. These lines traverse the edges of \(B_w\) in order to locate a cycle or deduce that no cycle exists along the chosen path (condition in line 16). This iterative structure takes at most \(|B_w|\) steps since all edges of \(B_w\) will be chosen, if \(B_w\) is a cycle itself. The condition in line 15 takes \(|B_w|\) steps since the array \(T\), of \(|B_w|\) elements at most, is checked for repetitions of vertices. Thus, in total, the iterative structure in lines 11 to 15 takes \(|B_w|^2\) steps in the worst case.

Then, depending on the condition check in line 16, one of the iterative structures in lines 17 to 20 and lines 23 to 26 respectively, will be executed. Both structures take \(|B_w|\) steps at most.

With respect to the non iterative structures in lines 5 to 36, we have the following operations:

- Simple arithmetic and assignment operations: they cost one step.
- The existential condition in line 6: we assume that we keep a one dimensional matrix of size \(n\) with the degrees of all vertices. Then this step is performed by looking at this array to locate a vertex, in \(B_w\), of degree other than 1. Thus, it takes \(|B_w|\) steps.
- The initialization in line 7: \(T\) has \(|B_w|\) non-zero values, at most, from the previous iteration and, thus, it takes that much time to initialize to 0.
- The selection operation in line 12: we simply look at the adjacency list of \(G\) to locate the next available vertex which is adjacent to the current vertex. This takes \(|B_w|\) steps.
- The condition in line 15: we look at the array \(T\) to locate a repetition of a vertex. This takes time \(|B_w|\).
- The inclusion test in 27: for each element of \(U\) we test whether it belongs to \(B_w^+\) or \(B_w^-\). This takes time \(|B_w|\) in total.
- The update in line 31: this deletes the edges in \(U\) from \(E\) and, thus, from \(B_w\). This requires \(|B_w|\) steps.

Summing up, the complexity of the algorithm is in the order of \(|Gr_A|\cdot \max_w |B_w|^3\).

From proposition \[3.5\] we know that \(|B_w| \leq 2(n - 2)\). Therefore the following corollary holds, with \(O(f(n))\) denoting functions of \(n\) bounded above by \(cf(n)\), for some constant \(c > 0\) and all \(n > n_0\), for some constant \(n_0\).
Corollary 3.7. The time complexity of Algorithm UGB is \( O(|Gr_A| \cdot n^3) \).

Remark 3.8. As it is evident from the proof of Theorem 3.6, the algorithm UGB decides in polynomial time whether a single element of the Graver basis of \( I_G \) belongs also in the \( U_A \). However, since the algorithm must test exhaustively all elements of the Graver basis for inclusion in the Universal Gröbner basis, the factor \( |Gr_A| \) is present in the final complexity figure. For some classes of graphs this factor is not prohibitively large, i.e. it is polynomial in \( n \). For instance, the class of graphs that consists of even cycles connected by paths where each cut vertex belongs to exactly two blocks has at most \( \frac{n}{4} \) cycles since every cycle has at least four edges. Therefore its Graver basis has cardinality \( |Gr_A| \leq \frac{n}{4} \). For such graphs, the complexity of the algorithm UGB is (from Corollary 3.7) \( O(n^4) \) which is a polynomial in the number \( n \) of vertices of the graph.

However, in many cases the size of the Graver basis appears to grow exponentially fast with \( n \). Thus, due to its large size, in general, there can be no polynomial time algorithm for computing the elements of the Graver basis of a general toric ideal. As an indication of the computational difficulty of this problem, in [4] J. De Loera, B. Sturmfels and R. Thomas showed, computationally, that the number of the primitive elements of \( I_{K_n} \) is 45570, where \( K_n \) is the complete graph on \( n \) vertices. In general, there are no (to the best of our knowledge) general tight upper bounds to the cardinality of the Graver basis of a graph since this cardinality depends on its structure. Only a rough idea for the size of the Graver basis of a toric ideal of a graph \( G \) can be obtained from the bound to the degrees of its elements (see proposition 3.5). However, the actual size of the Graver basis itself can be huge. For toric ideals of graphs, the problem of computing algorithmically the Graver basis reduces to finding the set of even closed walks of a corresponding graph \( G \), which is again a very large set. To the best of our knowledge, existing algorithmic techniques can only sample elements of the Graver bases in polynomial time (see [5]) or compute Graver bases by specialized mathematical software, such as 4ti2 (see [10]), for graphs with small Graver bases.

References

[1] W. Adams, P. Loustaunau, An Introduction to Gröbner Bases, Graduate Studies in Mathematics, Vol. 3 AMS, 1994.
[2] T. Becker, H. Kredel, V. Weispfenning, Gröbner Bases: A Computational Approach to Commutative Algebra, Graduate Texts in Mathematics, Springer, New York, (1993).
[3] D. Cox, J. Little, D. O’shea, Ideals, Varieties and Algorithms, Undergraduate Texts in Mathematics, Springer, New York, (1992).
[4] J. De Loera, B. Sturmfels and R. Thomas, Gröbner bases and triangulations of the second hypersimplex, Combinatorica 15 (3) (1995) 409–424.
[5] M. Ogawa, H. Haru, A. Takemura, Graver basis for an undirected graph and its application to testing the beta model of random graphs, Ann. Inst. Stat. Math. DOI 10.1007/s10463-012-0367-8.
[6] E. Reyes, Ch. Tatakis, A. Thoma, Minimal generators of toric ideals of graphs, Advances in Appl. Math. 48(1) (2012) 64-78.
[7] N. Schwartz, Stability of Gröbner Bases, J. Pure Appl. Algebra 53 (1988) 171–186.
[8] B. Sturmfels, Gröbner Bases and Convex Polytopes. University Lecture Series, No. 6 American Mathematical Society Providence, R.I. 1995.
[9] Ch. Tatakis, A. Thoma, On the universal Gröbner bases of toric ideals of graphs, J. Combin. Theory Ser. A 118 (2011) 1540-1548.
[10] 4ti2 team (2008). 4ti2a software package for algebraic, geometric and combinatorial problems on linear spaces. Available at http://www.4ti2.de
[11] R. Villarreal, Rees algebras of edge ideals, Comm. Algebra 23 (1995) 3513-3524.
[12] V. Weispfenning, Constructing universal Gröbner Bases, in Proceedings AAECC 5, Menorca, Springer Lecture Notes in Computer Science 356 (1987) 408–417.

DEPT. OF BUSINESS ADMINISTRATION, UNIVERSITY OF PATRAS AND COMPUTER TECHNOLOGY INSTITUTE AND PRESS - “DIOPHANTUS”, PATRA, GREECE
E-mail address: stamatiou@ceid.upatras.gr

MITILINI, P.O. BOX 13, MITILINI (LESVOS) 81100, GREECE
E-mail address: chtataki@cc.uoi.gr