

# Computing convex hulls by automata iteration

François Cantin<sup>1</sup>, Axel Legay<sup>2</sup>, and Pierre Wolper<sup>1</sup>

<sup>1</sup> Université de Liège, Institut Montefiore, Liège, Belgium  
{cantin,pw}@montefiore.ulg.ac.be

<sup>2</sup> Carnegie Mellon University, Computer Science Department, Pittsburgh, PA,  
alegay@cs.cmu.edu

**Abstract.** This paper considers the problem of computing the real convex hull of a finite set of  $n$ -dimensional integer vectors. The starting point is a finite-automaton representation of the initial set of vectors. The proposed method consists in computing a sequence of automata representing approximations of the convex hull and using extrapolation techniques to compute the limit of this sequence. The convex hull can then be directly computed from this limit in the form of an automaton-based representation of the corresponding set of real vectors. The technique is quite general and has been successfully implemented. Also, our result fits in a wider scheme whose objective is to improve the techniques for converting automata-based representation of constraints to formulas.

## 1 Introduction

Automata-based representations for sets of integer and real vectors have been a subject of growing interest in recent years [2, 4, 14, 19, 21]. While usually not optimal for specific problems, they provide much stronger generality and canonicity than other representations. For instance, in this context, combining real and integer constraints is very simple once the right framework has been set up [5]. The benefit of using automata-based representations for arithmetic sets could be even greater if one could, whenever appropriate, freely move between this and other representations such as explicit constraints. Going from constraints to automata has long been successfully studied [10, 3, 8], but going in the other direction is substantially more difficult. Nevertheless, it has been shown that it is possible [20] to construct constraint formulas from automata representing sets of integer vectors and that, under some restrictions, this can be done quite effectively [18].

One case that is not well handled though is that of finite sets of integer vectors. Indeed, imagine that a finite set of integers is represented by constraints and that an automaton representing this set is built from these. Since the set is finite, this acyclic automaton lacks the structure needed to construct the corresponding constraints. One is thus stuck with the automaton or with an enumerative representation of the set it defines, which is far from satisfactory. The work presented here was motivated by this problem with the idea of solving it along the following lines. The first step is to compute, as an automaton, a

minimal dense set of real-vectors that contains the finite set of integers. On this automaton, techniques similar to those of [18, 20] could then be applied to obtain constraints.

This paper proposes a solution for the first step in the form of a purely automata-based technique for computing the real convex hull (*i.e.* the convex hull over  $\mathbb{R}^n$ ) of a finite automaton-represented finite set of integers. Note that, beyond the motivation outlined above, this is also a worthwhile challenge of independent interest in the area of automata-based representations. In simple terms, our approach proceeds as follows. We start with an automata-based representation of a finite set of integer vectors. We then repeatedly apply a transformation to this automaton that adds to the set the vectors that are mid-way between those it includes. This yields an infinite sequence of automata-represented sets. The limit of this infinite sequence is then computed as an automaton, using the extrapolation-based techniques of [6]. This limit is not quite the convex closure since we prove that it will only contain convex combinations of the initial vectors with coefficients that are multiples of a negative power of 2. This limit thus needs to be “completed” in order to obtain the convex hull and we show that this can be done by computing its topological closure. Bar a technical point due to the fact that some reals have two encodings in our framework, the computation of the topological closure is quite an easy step. This being done, the closure is obtained.

The extrapolation-based techniques of [6], which have so far only been applied in the context of “regular model checking” [1, 9], are semi-algorithms that tackle the undecidable problem of computing the limit of an infinite sequence by extrapolating finite prefixes of the sequence. For the procedure above to work correctly, we thus depend on the result of the extrapolation being exact, which is not guaranteed a priori. Nevertheless, this can be checked as described in [6], but one interesting twist is that checking safety (enough is obtained) can be done much more easily (and just as correctly) after computing the topological closure. This is due to the fact that taking the topological closure yields an automaton that falls within an easier to handle class. Checking preciseness (nothing is added) with the techniques of [6] is probably not practical, but in the present situation one can exploit the properties of the extrapolation and make this check just as simple as the safety check.

Our approach has been implemented and the implementation has actually served as a guide to hone our results. The implementation has been tested and performs well, within the bounds allowed by the automata manipulations needed for the computation of the limit of the sequence of approximations. We certainly do not claim to outperform more traditional methods when they apply, our goal being to establish the basis of a different approach with interesting characteristics, performance gains not being part of our initial agenda. Also note that complexity analysis would not yield useful information since, at the heart of our approach, lies the extrapolation procedure which is only a semi-algorithm.

## 2 Automata-theoretic background

In this section we recall some automata-theoretic concepts used in the paper and give a brief description of the use of automata to encode sets of integer and real vectors.

### 2.1 Automata on infinite words

An infinite word (or  $\omega$ -word)  $w$  over an alphabet  $\Sigma$  is a mapping  $w : \mathbb{N} \rightarrow \Sigma$  from the natural numbers to  $\Sigma$ . The length- $k$  prefix of an infinite word  $w$ , i.e. the finite-word  $w(0), w(1), \dots, w(k-1)$ , will be denoted by  $\text{pref}_k(w)$ .

A *Büchi automaton* on infinite words is a five-tuple  $A = (Q, \Sigma, \delta, q_0, F)$ , where  $Q$  is a finite set of states,  $\Sigma$  is the input alphabet,  $\delta : Q \times \Sigma \rightarrow 2^Q$  is a *transition function* ( $\delta : Q \times \Sigma \rightarrow Q$  if the automaton is deterministic),  $q_0$  is the initial state, and  $F$  is a set of accepting states. A *run*  $\pi$  of a Büchi automaton  $A = (Q, \Sigma, \delta, q_0, F)$  on an  $\omega$ -word  $w$  is a mapping  $\pi : \mathbb{N} \rightarrow Q$  such that  $\pi(0) = q_0$  and for all  $i \geq 0$ ,  $\pi(i+1) \in \delta(\pi(i), w(i))$  (nondeterministic automata) or  $\pi(i+1) = \delta(\pi(i), w(i))$  (deterministic automata). Let  $\text{inf}(\pi)$  be the set of states that occur infinitely often in a run  $\pi$ . A run  $\pi$  is said to be *accepting* if  $\text{inf}(\pi) \cap F \neq \emptyset$ . An  $\omega$ -word  $w$  is *accepted* by a Büchi automaton if that automaton has some accepting run on  $w$ . The language  $L_\omega(A)$  of infinite words defined by a Büchi automaton  $A$  is the set of  $\omega$ -words it accepts.

We will also use the notion of *weak* automata [23]. Roughly speaking, a weak automaton is a Büchi automaton such that each of the strongly connected components of its graph contains either only accepting or only non-accepting states. Not all omega-regular languages can be accepted by weak deterministic Büchi automata, nor even by weak nondeterministic automata. However, there are algorithmic advantages to working with weak automata. Indeed, weak deterministic automata can be complemented simply by inverting their accepting and non-accepting states, while the complementation operation for Büchi automata requires intricate algorithms that not only are worst-case exponential, but are also hard to implement and optimize [26]. There exists a simple determinization procedure for weak automata [24], which produces Büchi automata that are deterministic, but not necessarily weak. A final advantage of weak deterministic Büchi automata is that they admit a normal form, which is unique up to isomorphism [22].

### 2.2 Automata-based representations of sets of integers and reals

In this section, we briefly introduce the representation of sets of integer and real vectors by finite automata. Details are only given for the case of real vectors, the case of integer vectors being a simplification of the former where automata on finite words replace automata on infinite words. A survey on this topic can be found in [8].

In order to make a finite automaton recognize numbers, one needs to establish a mapping between these and words. Our encoding scheme corresponds to the

usual notation for reals and relies on an arbitrary integer base  $r > 1$ . We encode a number  $x$  in base  $r$ , most significant digit first, by words of the form  $w_I \star w_F$ , where  $w_I$  encodes the integer part  $x_I$  of  $x$  as a finite word over  $\{0, \dots, r-1\}$ , the special symbol “ $\star$ ” is a separator, and  $w_F$  encodes the fractional part  $x_F$  of  $x$  as an infinite word over  $\{0, \dots, r-1\}$ . Negative numbers are represented by their  $r$ ’s complement. The length  $p$  of  $|w_I|$ , which we refer to as the *integer-part length* of  $w$ , is not fixed but must be large enough for  $-r^{p-1} \leq x_I < r^{p-1}$  to hold.

According to this scheme, each number has an infinite number of encodings, since their integer-part length can be increased unboundedly. In addition, the rational numbers whose denominator has only prime factors that are also factors of  $r$  have two distinct encodings with the same integer-part length. For example, in base 10, the number  $11/2$  has the encodings  $005 \star 5(0)^\omega$  and  $005 \star 4(9)^\omega$ , “ $\omega$ ” denoting infinite repetition. We call these respectively the *high* and *low* encodings and refer collectively to them as *dual* encodings.

To encode a vector of real numbers, we represent each of its components by words of identical integer-part length. This length can be chosen arbitrarily, provided that it is sufficient for encoding the vector component with the highest magnitude. An encoding of a vector  $\mathbf{x} \in \mathbb{R}^n$  can indifferently be viewed either as a  $n$ -tuple of words of identical integer-part length over the alphabet  $\{0, \dots, r-1, \star\}$ , or as a single word  $w$  over the alphabet  $\{0, \dots, r-1\}^n \cup \{\star\}$ .

Using an alphabet of size  $r^n + 1$  is clearly going to be problematic as soon as  $n$  starts to grow. The solution proposed in [5, 27] is to read the digits of the various components of the vector serially, in a round robin way, thus reducing the alphabet size to the perfectly manageable  $r + 1$ . This scheme is referred as the *serial encoding* as opposed to the *simultaneous encoding* in which the alphabet consists of tuples of digits.

Implementations obviously use the serial encoding, but the simultaneous encoding is convenient for presentation and proof purposes. The set of all the encodings of a vector  $\mathbf{v} \in \mathbb{R}^n$  is denoted by  $W(\mathbf{v}, n)$ . This definition directly generalizes to sets of vectors.

Real vectors being encoded by infinite words, a set of vectors can be represented by an infinite-word automaton accepting the corresponding encodings. Since a real vector has an infinite number of possible encodings, we have to choose which of these the automata will recognize. A natural choice is to accept all encodings. This leads to the following definition.

**Definition 1.** *Let  $n > 0$  and  $r > 1$  be integers. A base- $r$   $n$ -dimension serial Real Vector Automaton (RVA) [7] is a Büchi automaton  $\mathcal{A} = (Q, \Sigma, \delta, Q_0, F)$  over the alphabet  $\Sigma = \{0, \dots, r-1\} \cup \{\star\}$ , such that (1) Every word accepted by  $\mathcal{A}$  is a serial encoding in base  $r$  of a vector in  $\mathbb{R}^n$ , and (2) For every vector  $\mathbf{x} \in \mathbb{R}^n$ ,  $\mathcal{A}$  accepts either all the encodings of  $\mathbf{x}$  in base  $r$ , or none of them.*

An RVA is said to *represent* the set of vectors encoded by the words that belong to its accepted language. In [5], it is shown that if the set represented by the RVA can be defined in the first-order theory of linear constraints, then this

RVA can be transformed into an equivalent weak deterministic Büchi automata. If not explicitly mentioned, we assume that the RVAs we manipulate are minimal weak deterministic Büchi automata. Also, since our implementation works with a base 2 representation, we will present all our results in this context, knowing that they can be generalized to other bases.

### 3 Convex Hulls and Topological Concepts

We recall a few notations and definitions that are used throughout the paper.

Let  $\mathbb{Z}$ ,  $\mathbb{Q}$ , and  $\mathbb{R}$  be respectively the sets of integers, rational, and reals, and let  $\mathbb{Z}^n$ ,  $\mathbb{Q}^n$ , and  $\mathbb{R}^n$  denote the usual  $n$ -dimensional Euclidean vector spaces. Vectors are written in boldface, e.g.  $\mathbf{x}$ , and scalars without emphasis, e.g.  $a$ . The  $i$ th component of a vector  $\mathbf{x} \in \mathbb{R}^n$  is denoted by  $\mathbf{x}[i]$ . We say that a set  $E \subseteq \mathbb{R}^n$  is *convex* iff for each  $\mathbf{x}_1, \mathbf{x}_2 \in E$ , we have  $\{\alpha\mathbf{x}_1 + (1 - \alpha)\mathbf{x}_2 \mid \alpha \in [0, 1]\} \subseteq E$ . We will also use the following usual definitions.

**Definition 2.** Given a set  $E \subseteq \mathbb{R}^n$ , the convex hull of  $E$  is the set  $\text{Conv}(E) \subseteq \mathbb{R}^n$  defined by

$$\text{Conv}(E) = \{\mathbf{x} \mid \exists \mathbf{x}_1, \dots, \mathbf{x}_k \in E \exists \lambda_1, \dots, \lambda_k \in [0, 1] \mathbf{x} = \sum_{i=1}^k \lambda_i \mathbf{x}_i \wedge \sum_{i=1}^k \lambda_i = 1\}$$

The *Euclidean distance* between two vectors  $\mathbf{x}, \mathbf{x}' \in \mathbb{R}^n$ , denoted by  $|\mathbf{x} - \mathbf{x}'|$  is the real number  $\sqrt{\sum_{i=1}^n (\mathbf{x}[i] - \mathbf{x}'[i])^2}$ . The *open ball* centered in  $\mathbf{x} \in \mathbb{R}^n$  with a radius  $\epsilon > 0$  is the subset  $B_{(\mathbf{x}, \epsilon)} = \{\mathbf{x}' \mid |\mathbf{x} - \mathbf{x}'| < \epsilon\}$ . A set  $E \subseteq \mathbb{R}^n$  is said to be *open* if for any  $\mathbf{x} \in E$  there exists  $\epsilon > 0$  such that  $B_{(\mathbf{x}, \epsilon)} \subseteq E$ . A *closed* set  $E$  is a subset of  $\mathbb{R}^n$  such that  $\mathbb{R}^n \setminus E$  is an open set. A *compact* set in  $\mathbb{R}^n$  is a bounded and closed set. We use the concept of *topological closure* of a set.

**Definition 3.** Given a set  $E \subseteq \mathbb{R}^n$ , the topological closure  $TC(E)$  of  $E$  is the smallest closed set that contains  $E$ .

When dealing with infinite words, we will be working with the topology on words induced by the distance defined by

$$d(w, w') = \begin{cases} \frac{1}{|\text{common}(w, w')| + 1} & \text{if } w \neq w' \\ 0 & \text{if } w = w', \end{cases}$$

where  $\text{common}(w, w')$  denotes the longest common prefix of  $w$  and  $w'$ . Notice that, among words that validly encode vectors, words that are topologically close encode vectors that are close according to the Euclidean distance, the reverse also being true except for the cases where dual encodings can appear.

## 4 Computing convex hulls

In this section, we describe a technique to compute the convex hull over  $\mathbb{R}^n$  of a finite set  $E = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_k\}$  defined over  $\mathbb{Z}^n$ .

The technique proceeds by constructing a sequence of approximations of the convex hull by adding the vectors that are mid-way between those obtained so far. This is quite an obvious way to proceed, but in order to exploit it, we need to formalize its exact properties. We use the following definitions.

**Definition 4.** *The median sequence of  $E$  is the infinite sequence  $E_0, E_1, E_2, \dots$  such that (1)  $E_0 = E$  and (2)  $E_{i+1} = E_i \cup \{(\mathbf{x}_1 + \mathbf{x}_2)/2 \mid \mathbf{x}_1, \mathbf{x}_2 \in E_i\}$  for each  $i \in \mathbb{N}$ .*

The *limit* of the median sequence of  $E$ , denoted by  $E^*$ , is defined by  $\bigcup_{i=0}^{\infty} E_i$ . It is easy to see that each vector  $\mathbf{v}$  of  $E^*$  is also a vector of  $\text{Conv}(E)$ . However,  $E^*$  is not the complete convex hull, but can be characterized using the following definition.

**Definition 5.** *The 2-chopped convex hull of a finite subset  $E = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_k\}$  of  $\mathbb{Z}^n$  is the maximal subset  $\text{Conv}_{2^*}(E)$  of  $\text{Conv}(E)$ , where for each  $\mathbf{v} \in \text{Conv}_{2^*}(E)$ ,  $\mathbf{v} = \sum_{i=1}^k \lambda_i \mathbf{x}_i$  with  $\lambda_i \in [0, 1]$ ,  $\sum_{i=1}^k \lambda_i = 1$ , and  $\lambda_i = \frac{k_i}{2^{m_i}}$  for  $k_i, m_i \in \mathbb{N}$  and  $i \in [1, \dots, k]$ .*

**Theorem 1.** *For any finite subset  $E = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_k\}$  of  $\mathbb{Z}^n$ , the limit of its median sequence and its 2-chopped convex hull coincide, i.e.  $E^* = \text{Conv}_{2^*}(E)$ .*

Even though the 2-chopped convex hull of a set  $E$  is not quite its real convex hull, it contains vectors that are arbitrarily close to any element of the full convex closure.

**Lemma 1.** *For each  $\mathbf{v} \in \text{Conv}(E)$  and  $\epsilon > 0$ , there exists  $\mathbf{v}' \in \text{Conv}_{2^*}(E)$  such that  $|\mathbf{v} - \mathbf{v}'| \leq \epsilon$ .*

From Lemma 1 it follows that the convex hull of  $E$  is included in the topological closure of its 2-chopped hull. The following theorem states that these two sets coincide.

**Theorem 2.** *For any finite subset  $E = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_k\}$  of  $\mathbb{Z}^n$ , we have that  $TC(\text{Conv}_{2^*}(E)) = \text{Conv}(E)$ .*

Computing the real convex hull of a finite set of integer vectors can thus be reduced to compute the topological closure of the limit of its median sequence. We now investigate how to compute  $\text{Conv}_{2^*}(E)$  and  $TC(E)$  for a set  $E$  described by an *RVA*.

## 5 Algorithmic issues

We consider a finite subset  $E = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_k\}$  of  $\mathbb{Z}^n$  that is represented by a (weak deterministic) RVA  $A_E$ . Our goal is to compute an RVA that represents the convex hull over  $\mathbb{R}^n$  of  $E$ . According to the results in Section 4, this can be done by computing an RVA  $A_{E^*}$  representing the limit  $E^*$  of the median sequence of  $E$ , and then computing an RVA representing the topological closure of  $E^*$ . We now show how these two problems can be tackled by automata-based semi-algorithms.

### 5.1 Computing an RVA for the 2-chopped Hull

**Computing the elements of the median sequence** We notice that since  $E$  is finite and represented by a weak deterministic RVA, each element in its median sequence can also be represented in the same way. Indeed, computing  $E_{i+1}$  from  $E_i$  can be done by simple operations (Cartesian product, projection, union, intersection) on the automata representations that preserve their weak-deterministic nature (see [5]).

**Computing the limit of the median sequence** Computing  $A_{E^*}$  amounts to computing the limit of an infinite sequence of weak deterministic automata. To finitely compute this limit, we obviously need some form of “speed-up” technique. We will use the extrapolation-based technique proposed in [6]. A rough description of the technique is as follows. The technique proceeds by comparing successive automata in a prefix of the sequence, trying to identify the difference between these in the form of an “increment”, and extrapolating the repetition of this increment by adding loops to the last automaton of the prefix. If the extrapolation is *correct*, then the limit is computed, else, one has to lengthen the prefix and restart the extrapolation process. Checking correctness of the extrapolation is a non trivial procedure whose description is, for technical reasons, postponed to Section 5.3. The technique has been implemented in a tool called T(O)RMC [25]. The tool relies on the LASH package [17] for automata manipulation procedures, but implements the specific algorithms given in [6]. There is no guarantee that T(O)RMC will produce a result since the general problem of computing the limit of a sequence of automata is undecidable.

It is worth mentioning that the automata produced by T(O)RMC are weak, but not necessarily deterministic [6]. Furthermore, if one tries to determinize these automata, one might end up combining accepting and non accepting connected components, which leads to an automaton that is not weak. This situation actually occurred systematically in our experiment, which is not surprising since the 2-chopped convex hull of a set of integer vectors is not definable in  $\langle \mathbb{R}, +, \leq, Z \rangle$  and thus falls outside the guaranteed reach of weak deterministic automata given in [5].

## 5.2 Computing the topological closure of an RVA-represented set

In this section, we explicitly consider RVAs that may not be weak deterministic. Consider a set  $E \subseteq \mathbb{R}^n$  represented by an RVA  $A_E$ . Our goal is to compute an RVA  $A_{TC(E)}$  that represents the topological closure of  $E$ . The intuition behind the computation is that we need to add to the language accepted by  $A_E$ , all words that are arbitrarily close to words of this language. This is fairly straightforward to do since we only need to add words that have arbitrarily long common prefixes with accepted words. A simple step to do this is to make accepting all states of the fractional part of the automaton. Of course, this will compute the topological closure within the topology on infinite words, but this also almost computes the vector Euclidean topological closure as it shown by the following result.

**Theorem 3.** *Let  $A_E$  be an RVA representing a vector set  $E$ . Let  $\overline{A}_E$  be  $A_E$  with all states of its fractional part made accepting. For each vector  $\mathbf{v} \in \mathbb{R}^n$ ,  $W(\mathbf{v}, n) \cap L(\overline{A}_E) \neq \emptyset$  if and only if  $\mathbf{v} \in TC(E)$ .*

Theorem 3 guarantees that  $\overline{A}_E$  contains at least one encoding for each vector in  $TC(E)$ . However the automaton  $\overline{A}_E$  is not necessarily  $A_{TC(E)}$ . Indeed, there is no guarantee that  $\overline{A}_E$  will contain *all* the encodings of each vector included in the topological closure.

*Example 1.* Assume that  $A_E$  is the RVA representing the 2-chopped hull of the set  $E = \{(0, 0), (6, 3)\}$ . Here,  $\overline{A}_E$  is not a proper RVA. Indeed, the vector  $(2, 1)$  belongs to the topological closure of  $A_E$ , but the infinite word  $w = 001000\star(01)^\omega$  that corresponds to the high encoding of 2 and the low encoding of 1 is never added.

We thus need an extra step that adds all missing encodings. To do this, we use the fact that an automaton that recognizes words that are dual encodings of the same numbers can be built with simple automata-based operations (see [11] for the detailed algorithm).

## 5.3 Correctness criterion

After having constructed the extrapolation  $A_E^*$  of a finite sequence  $A_E^{i_1}, A_E^{i_2}, \dots, A_E^{i_k}$  of automata representing elements in the median sequence of a set  $E$ , it remains to check whether it accurately corresponds to what we really intend to compute, i.e.,  $A_{E^*}$ . This is done by first checking that the extrapolation is *safe*, in the sense that it captures all words accepted by  $A_{E^*}$  ( $L(A_{E^*}) \subseteq L(A_E^*)$ ), and then checking that it is *precise*, i.e. that it accepts no more words than  $A_{E^*}$  ( $L(A_E^*) \subseteq L(A_{E^*})$ ). To lighten the presentation, we will often use the notations and operations defined for sets of vectors directly on the automata that represent them. As an example, given an RVA  $A$ ,  $Conv(A)$  is the RVA that represents the convex hull of the set represented by  $A$ .

**Safety** We first investigate how to check whether  $A_E^*$  is safe. The idea is simply to perform one more mid-point adding step on  $A_E^*$  and to check that this does not change the accepted language. Given a set  $E$ , let  $C_2(E)$  be the set  $\{\mathbf{y} \mid \mathbf{y} = (\mathbf{x}_1 + \mathbf{x}_2)/2 \mid \mathbf{x}_1, \mathbf{x}_2 \in E\}$ . We have the following theorem.

**Theorem 4.** *Let  $A_E^*$  and  $A_{E^*}$  be respectively the extrapolation of a median automata sequence for a set  $E$  and a representation of the actual limit of this sequence. We have that, if  $L(C_2(A_E^*)) \subseteq L(A_E^*)$ , then  $L(A_{E^*}) \subseteq L(A_E^*)$ .*

The required computation step is thus to check that  $L(C_2(A_E^*)) \subseteq L(A_E^*)$ . This is simple except for the fact that, the result of the extrapolation is representable by an automaton which is weak but not necessarily deterministic (see Section 5.1), and hence testing inclusion requires to complement a Büchi automaton. The problem can be solved by first applying the topological closure step to  $A_E^*$  and then performing the safety check given by Lemma 4. It is easy to see that doing this has no impact on the result of the test. However it has an impact on its efficiency since the strongly connected component added by T(O)RM are made uniformly accepting status by the procedure that computes the topological closure. This ensures that we only need to complement weak deterministic automata.

**Preciseness** Checking preciseness could be performed with the techniques proposed in [6]. However, this solution (which involves counter automata) is computationally demanding and not really practical. In the present situation, one can however propose a much more efficient scheme that exploits the properties of the extrapolation. Due to space limitation, we report the details of this contribution to Appendix E, and only present the main result.

**Definition 6.** *Let  $E \in \mathbb{R}$  be a convex set. The set of extreme points of  $E$ , denoted  $S(E)$ , is defined as  $\{\mathbf{x} \in E \mid (\neg \exists (\mathbf{x}_1, \mathbf{x}_2) \in E)(\mathbf{x}_1 \neq \mathbf{x}_2 \wedge \mathbf{x} = (\mathbf{x}_1 + \mathbf{x}_2)/2)\}$ .*

By extension we will also use notation  $S(E)$  on automata representing vector sets. We now present our preciseness check. Instead of checking whether  $L(A_E^*) \subseteq L(A_{E^*})$ , we check  $L(TC(A_E^*)) \subseteq L(Conv(A_E))$ . This is enough to ensure that we do not compute an overapproximation of the hull.

**Theorem 5.** *Let  $A_E^*$  be an RVA that represents a safe extrapolation of the limit of the median sequence of a finite set of integer vectors represented by the RVA  $A_E$ . If  $L(S(TC(A_E^*))) \subseteq L(A_E)$ , then  $L(TC(A_E^*)) \subseteq L(Conv(A_E))$ .*

In summary, to check the preciseness of an RVA  $A_E^*$  that represents a safe extrapolation of the limit of the median sequence of a finite set  $E \subseteq \mathbb{Z}^n$ , we first compute an RVA  $TC(A_E^*)$  for the topological closure of the set represented by  $A_E^*$ . We then compute an automaton for  $S(TC(A_E^*))$ , which is easily done by computing the difference between  $TC(A_E^*)$  and  $C_2(TC(A_E^*))$ . Finally, one checks whether the language of the resulting automaton is included in the one of  $A_E$ . Again, all complementation operations are only applied to weak deterministic Büchi automata.

## 6 Infinite Sets

It is worth mentioning that our results do not extend as such to the computation of the real convex hull of an infinite set of integer vectors. Indeed, by relying on the computation of a topological closure, our methodology produces convex hulls which are closed sets. However there are infinite sets of integer vectors whose convex hull is not closed.

*Example 2.* Consider the infinite set  $E$  given by  $\{(x, y) \in \mathbb{Z}^2 \mid (y = x + 1) \wedge (y \geq 0)\} \cup \{(0, 0)\}$ . The convex hull of  $E$  is given by  $\text{Conv}(E) = \{(x, y) \in \mathbb{R}^2 \mid (y \leq x + 1) \wedge (y \geq 0) \wedge (y > x)\}$ , which is not a closed set. If we apply our technique to  $E$ , we will obtain the set  $\{(x, y) \in \mathbb{R}^2 \mid (y \leq x + 1) \wedge (y \geq 0) \wedge (y \geq x)\}$ , that is a convex overapproximation of  $\text{Conv}(E)$ .

Observe also that when working with infinite sets, we cannot check the preciseness of the extrapolation with the technique proposed in Section 5.3. Indeed, this check relies on Krein-Milman's theorem, which only applies to bounded sets.

Consequently, when working with infinite sets, the best we can produce is a convex overapproximation of the real convex hull. This does not prevent the result to be usefull in many practical applications such as the verification of linear hybrid systems by means of convex approximations [16].

## 7 A brief note on the experimental results

The approach presented in this paper has been tested on several examples using a prototype implementation that relies on T(O)RMC. We computed the convex hull over  $\mathbb{R}^n$  of finite convex sets in  $\mathbb{Z}^n$ , of the difference/union between finite convex sets in  $\mathbb{Z}^n$  and of arbitrary finite sets of points in  $\mathbb{Z}^n$ . Some preliminary results are reported in Appendix F. Those experiments validate the fact that our approach performs well for which the representation by automata remains manageable.

## 8 Conclusion and Related Work

This paper proposes a methodology to compute the real convex hull of a finite automaton-represented finite set of integers. Our result fits in a wider scheme that is to improve and generalize the techniques presented in [18, 20].

Computing convex hulls is of course a well studied problem of independent interest. There are quite a few known techniques for computing convex hulls of a set of vectors in a non automata-theoretic setting. Among these a long series of algorithms specialized to the 2D and 3D case and widely used and studied in computational geometry [13]. Algorithms for the general case (any dimensions) have also been studied [12]. All those algorithms, which are definitively more efficient than an automata-based approach, require an enumeration of the set, which we avoid here. In [15], Finkel and Leroux show that the convex hull of a

(possibly infinite) set of integer vector represented by an automaton is a computable polyhedron. The algorithm in [15] can be applied to infinite sets<sup>3</sup> and is guaranteed to terminate. On the other hand, this algorithm, may require to enumerate the set represented by the automaton and is restricted to work in  $\mathbb{Z}^n$ .

Though we can claim originality with respect to the approach and techniques used, we do not, at this point, claim efficiency with respect to the algorithms used in computational geometry, nor generality with respect to, for instance, [15]. Our contribution thus stands in a middle ground where it offers generality coming from the automata representation, but is limited by the incompleteness of the extrapolation technique and, compared to more theoretical work, is backed by an implementation that fits in a wider scheme. In other words, there is no measure on which we can claim to beat all other approaches, but what we propose has at least some strong point with respect to each of the alternatives.

## Acknowledgement

We thank Pierre Mathonet and Michel Rigo from the mathematics department of the University of Liège for answering many questions related to this work. We are also grateful to Felix Klaedtke of ETH Zurich for providing insightful comments that have helped improving this work.

## References

1. P. A. Abdulla, A. Bouajjani, B. Jonsson, and M. Nilsson. Handling global conditions in parameterized system verification. In *Proc of CAV*, volume 1633 of *LNCS*, pages 134–145. Springer, 1999.
2. C. Bartzis and T. Bultan. Construction of efficient bdds for bounded arithmetic constraints. In *Proc of TACAS*, volume 2619 of *LNCS*, pages 394–408. Springer, 2003.
3. B. Boigelot. *Symbolic Methods for Exploring Infinite State Spaces*. Collection des publications de la Faculté des Sciences Appliquées de l’Université de Liège, Liège, Belgium, 1999.
4. B. Boigelot and F. Herbreteau. The power of hybrid acceleration. In *Proc of CAV*, volume 4144 of *LNCS*, pages 438–451. Springer, 2006.
5. B. Boigelot, S. Jodogne, and P. Wolper. An effective decision procedure for linear arithmetic over the integers and reals. *ACM Transactions on Computational Logic*, 6(3):614–633, 2005.
6. B. Boigelot, A. Legay, and P. Wolper. Omega-regular model checking. In *Proc of TACAS*, volume 2988 of *LNCS*, pages 561–575. Springer, 2004.
7. B. Boigelot, S. Rassart, and P. Wolper. On the expressiveness of real and integer arithmetic automata (extended abstract). In *Proc of ICALP*, volume 1443 of *LNCS*, pages 152–163. Springer, 1998.
8. B. Boigelot and P. Wolper. Representing arithmetic constraints with finite automata: An overview. In *Proc of ICLP*, volume 2401 of *LNCS*, pages 1–19. Springer, 2002.

---

<sup>3</sup> Recall that our technique can also be used to compute a possibly overapproximation of the convex hull of an infinite set

9. A. Bouajjani, B. Jonsson, M. Nilsson, and T. Touili. Regular model checking. In *Proc of CAV*, volume 1855 of *LNCS*, pages 403–418. Springer-Verlag, 2000.
10. A. Boudet and H. Comon. Diophantine equations, presburger arithmetic and finite automata. In *Proc of ICALP*, volume 1059 of *LNCS*, pages 30–43. Springer, 1996.
11. F. Cantin. Techniques d’extrapolation d’automates: Application au calcul de la fermeture convexe. Master’s thesis, University of Liège, Belgium, 2007.
12. B. Chazelle. An optimal convex hull algorithm in any fixed dimension. *Discrete & Computational Geometry*, 10:377–409, 1993.
13. B. Chazelle and J. Matousek. Derandomizing an output-sensitive convex hull algorithm in three dimensions. *Computational Geometry*, 5:27–32, 1995.
14. J. Eisinger and F. Klaedtke. Don’t care words with an application to the automata-based approach for real addition. In *Proc of CAV*, volume 4144 of *LNCS*, pages 67–80. Springer, 2006.
15. A. Finkel and J. Leroux. The convex hull of a regular set of integer vectors is polyhedral and effectively computable. *Information Processing Letter*, 96(1):30–35, 2005.
16. N. Halbwachs, Y. Proy, and P. Roumanoff. Verification of real-time systems using linear relation analysis. *Formal Methods in System Design*, 11(2):157–185, 1997.
17. The Liège Automata-based Symbolic Handler (LASH). Available at <http://www.montefiore.ulg.ac.be/~boigelot/research/lash/>.
18. L. Latour. From automata to formulas: Convex integer polyhedra. In *Proc of LICS*, pages 120–129. IEEE Computer Society, 2004.
19. J. Leroux. *Algorithmique de la vérification des systèmes à compteurs. Approximation et accélération. Implémentation de l’outil FAST*. PhD thesis, LSV Cachan, 2004.
20. J. Leroux. A polynomial time presburger criterion and synthesis for number decision diagrams. In *Proc of LICS*, pages 147–156. IEEE Computer Society, 2005.
21. J. Leroux and G. Sutre. Flat counter automata almost everywhere! In *Proc of ATVA*, volume 3707 of *LNCS*, pages 489–503. Springer, 2005.
22. C. Löding. Efficient minimization of deterministic weak  $\omega$ -automata. *Information Processing Letters*, 79(3):105–109, 2001.
23. D. E. Muller, A. Saoudi, and P. E. Schupp. Alternating automata, the weak monadic theory of the tree and its complexity. In *Proc of ICALP*, pages 275–283, Rennes, 1986. Springer-Verlag.
24. S. Safra. Exponential determinization for  $\omega$ -automata with strong-fairness acceptance condition. In *Proc of POPL*, Victoria, May 1992.
25. The T(O)RMC toolset. Available at <http://www.montefiore.ulg.ac.be/~legay/TORMC/index-tormc.html>.
26. M. Vardi. The büchi complementation saga. In *Proc of STACS*, LNCS. Springer, 2007. to appear.
27. P. Wolper and B. Boigelot. On the construction of automata from linear arithmetic constraints. In *Proc of TACAS*, volume 1785 of *LNCS*, pages 1–19. Springer, 2000.

## A Proof of Theorem 1

We use the following definition

**Definition 7.** A 2-term  $t$  of  $E \subset \mathbb{R}^n$  is either a vector of  $E$ , or an expression of the form  $(t_1 + t_2)/2$ , where  $t_1$  and  $t_2$  are two 2-term. The depth of  $t$ , denoted by  $d(t)$ , is 0 if  $t \in E$ , and  $\max(d(t_1), d(t_2)) + 1$  otherwise.

We now give the intuition behind the proof of Theorem 1.

*Proof.* We consider the two directions of the equivalence.

- We have  $E^* \subset \text{Conv}_{2^*}(E)$ . Indeed, by construction, each vector  $\mathbf{v} \in E^*$  can be expressed as a 2-term of  $E$ . Moreover, a 2-term  $t$  can be rewritten as an expression of the form

$$e = a_1 \mathbf{x}_1 + \cdots + a_k \mathbf{x}_k$$

with (1)  $\forall (1 \leq i \leq k) [(a_i \geq 0) \wedge (\exists (k_i, m_i \in \mathbb{N}) (a_i = \frac{k_i}{2^{m_i}}))]$  and (2)  $\sum_{i=1}^k a_i = 1$ . (1) is obvious by construction and (2) can easily be shown by induction on the depth of  $t$ .

- We have  $\text{Conv}_{2^*}(E) \subset E^*$ . Indeed, it is easy to see that each vector of  $\text{Conv}_{2^*}(E)$  can be rewritten as a 2-term of  $E$ . Moreover, a 2-term of depth  $i$  is, by construction, included in all  $E_j$  for  $j \geq i$ .

## B Proof of Lemma 1

We use the following lemma.

**Lemma 2.** Consider  $E = \{\mathbf{x}_1, \dots, \mathbf{x}_k\}$ , a finite set of vectors of  $\mathbb{R}^n$ . Let  $\mathbf{v} = \sum_{i=1}^k \lambda_i \mathbf{x}_i$ ,  $\mathbf{v}' = \sum_{i=1}^k \lambda'_i \mathbf{x}_i$ , and  $x_{\max} = \max_{i,j} (|\mathbf{x}_i[j]|)$  with  $i \in [1, k]$  and  $j \in [1, n]$ . For each  $\epsilon > 0$ , if  $\forall (1 \leq i \leq k) \exists (\epsilon_i > 0) |\lambda_i - \lambda'_i| \leq \epsilon_i$  such that  $\sum_{i=1}^k \epsilon_i \leq \frac{\epsilon}{\sqrt{n} x_{\max}}$ , then  $|\mathbf{v} - \mathbf{v}'| \leq \epsilon$ .

*Proof.*

$$\begin{aligned} & \forall i \exists \epsilon_i ( (|\lambda_i - \lambda'_i| \leq \epsilon_i) \wedge (\sum_{i=1}^k \epsilon_i \leq \frac{\epsilon}{\sqrt{n} x_{\max}}) ) \\ & \Leftrightarrow \sum_{i=1}^k |\lambda_i - \lambda'_i| \leq \frac{\epsilon}{\sqrt{n} x_{\max}} \\ & \Leftrightarrow \sqrt{\left( \sum_{i=1}^k |\lambda_i - \lambda'_i| \right)^2} \leq \frac{\epsilon}{\sqrt{n} x_{\max}} \\ & \Leftrightarrow \sqrt{n} x_{\max} \sqrt{(|\lambda_1 - \lambda'_1| + \cdots + |\lambda_k - \lambda'_k|)^2} \leq \epsilon \end{aligned}$$

$$\begin{aligned}
&\Leftrightarrow \sqrt{n x_{max}^2 (|\lambda_1 - \lambda'_1| + \dots + |\lambda_k - \lambda'_k|)^2} \leq \epsilon \\
&\Leftrightarrow \sqrt{n (|\lambda_1 - \lambda'_1| x_{max} + \dots + |\lambda_k - \lambda'_k| x_{max})^2} \leq \epsilon \\
&\Leftrightarrow [ (|\lambda_1 - \lambda'_1| x_{max} + \dots + |\lambda_k - \lambda'_k| x_{max})^2 + \dots \\
&\quad + (|\lambda_1 - \lambda'_1| x_{max} + \dots + |\lambda_k - \lambda'_k| x_{max})^2 ]^{\frac{1}{2}} \leq \epsilon \tag{1}
\end{aligned}$$

by Minkowski, for all  $1 \leq i \leq n$ ,

$$|(\lambda_1 - \lambda'_1) \mathbf{x}_1[i] + \dots + (\lambda_k - \lambda'_k) \mathbf{x}_k[i]| \leq |\lambda_1 - \lambda'_1| x_{max} + \dots + |\lambda_k - \lambda'_k| x_{max}$$

Therefore, we have

$$\begin{aligned}
(1) &\Rightarrow [ ((\lambda_1 - \lambda'_1) \mathbf{x}_1[1] + \dots + (\lambda_k - \lambda'_k) \mathbf{x}_k[1])^2 + \dots \\
&\quad + ((\lambda_1 - \lambda'_1) \mathbf{x}_1[n] + \dots + (\lambda_k - \lambda'_k) \mathbf{x}_k[n])^2 ]^{\frac{1}{2}} \leq \epsilon \\
&\Leftrightarrow [ ((\lambda_1 \mathbf{x}_1[1] + \dots + \lambda_k \mathbf{x}_k[1]) - (\lambda'_1 \mathbf{x}_1[1] + \dots + \lambda'_k \mathbf{x}_k[1]))^2 + \dots \\
&\quad + ((\lambda_1 \mathbf{x}_1[n] + \dots + \lambda_k \mathbf{x}_k[n]) - (\lambda'_1 \mathbf{x}_1[n] + \dots + \lambda'_k \mathbf{x}_k[n]))^2 ]^{\frac{1}{2}} \leq \epsilon \\
&\Leftrightarrow \sqrt{(\mathbf{v}[1] - \mathbf{v}'[1])^2 + \dots + (\mathbf{v}[n] - \mathbf{v}'[n])^2} \leq \epsilon \\
&\Leftrightarrow |\mathbf{v} - \mathbf{v}'| \leq \epsilon
\end{aligned}$$

We now prove Lemma 1.

*Proof.* We recall that  $E = \{\mathbf{x}_1, \dots, \mathbf{x}_k\}$ . We define  $x_{max} = \max_{i,j} (|\mathbf{x}_i[j]|)$  with  $i \in [1, k]$  and  $j \in [1, n]$ . For each  $\mathbf{v} \in \text{Conv}(E)$  and each  $\epsilon > 0$ , we build a vector  $\mathbf{v}' \in \text{Conv}_{2^*}(E)$  with  $|\mathbf{v} - \mathbf{v}'| \leq \epsilon$ . This amounts to assign a value to each  $\lambda'_i$ . This assignation is direct if  $\mathbf{v} \in \text{Conv}_{2^*}(E)$ .

Assume now that  $\mathbf{v} \notin \text{Conv}_{2^*}(E)$ . By hypothesis, we have

$$\begin{aligned}
- \mathbf{v} &= \sum_{i=1}^k \lambda_i \mathbf{x}_i, \text{ where } \sum_{i=1}^k \lambda_i = 1 \text{ and } \forall (1 \leq i \leq k) \lambda_i \geq 0. \\
- \mathbf{v}' &= \sum_{i=1}^k \lambda'_i \mathbf{x}_i, \text{ where } \sum_{i=1}^k \lambda'_i = 1 \text{ and } \forall (1 \leq i \leq k) [\lambda'_i \geq 0 \wedge \exists (k_i, m_i \in \mathbb{N}) (\lambda'_i = \frac{k_i}{2^{m_i}})].
\end{aligned}$$

By Lemma 2, if  $\forall (1 \leq i \leq k) \exists (\epsilon_i > 0) |\lambda_i - \lambda'_i| \leq \epsilon_i$  where  $\sum_{i=1}^k \epsilon_i \leq \frac{\epsilon}{\sqrt{n x_{max}}}$ , then  $|\mathbf{v} - \mathbf{v}'| \leq \epsilon$ .

Consider  $l \in \mathbb{N}$  with  $k 2^{-l} \leq \frac{\epsilon}{\sqrt{n x_{max}}}$ . For each  $1 \leq i \leq k$ , we define  $\lambda_{i_1}$  by truncating the 2's encoding of  $\lambda_i$  after the  $l$  first bits of its fractional part. It is easy to

see that  $\forall(1 \leq i \leq k) |\lambda_i - \lambda_{i_1}| \leq 2^{-l}$ . For each  $1 \leq i \leq k$ , let  $\lambda_{i_2} = \lambda_i - \lambda_{i_1} \leq 2^{-l}$ . Since  $\sum_{i=1}^k \lambda_{i_1}$  is a multiple of  $2^{-l}$ ,  $\sum_{i=1}^k \lambda_{i_2} = 1 - \sum_{i=1}^k \lambda_{i_1}$  is also a multiple of  $2^{-l}$ .

Define  $d \in \mathbb{N}$  as follows

$$d = \frac{\sum_{i=1}^k \lambda_{i_2}}{2^{-l}}$$

Since  $\forall(1 \leq i \leq k) \lambda_{i_2} \leq 2^{-l}$ , we have  $d \leq k$ . For each  $(1 \leq i \leq k)$ , we define  $\lambda'_i$  as follows:

$$\lambda'_i = \begin{cases} \lambda_{i_1} + 2^{-l} & \text{if } 1 \leq i \leq d \\ \lambda_{i_1} & \text{otherwise.} \end{cases}$$

We have

- $\forall(d < i \leq k) |\lambda_i - \lambda'_i| \leq 2^{-l}$ , and
- $\forall(1 \leq i \leq d) |\lambda_i - \lambda'_i| \leq |\lambda_i - (\lambda_i + 2^{-l})| = 2^{-l}$ .

Consequently,  $\forall(1 \leq i \leq k) |\lambda_i - \lambda'_i| \leq 2^{-l}$ , and by Lemma 2,  $|\mathbf{v} - \mathbf{v}'| \leq \epsilon$ .

To conclude, observe that

- $\forall(1 \leq i \leq k) \exists(k_i, m_i \in \mathbb{N}) \lambda'_i = \frac{k_i}{2^{m_i}}$ ,
- $\sum_{i=1}^k \lambda'_i = \sum_{i=1}^k \lambda_{i_1} + d2^{-l} = \sum_{i=1}^k \lambda_{i_1} + \sum_{i=1}^k \lambda_{i_2} = 1$ , and
- $\forall(1 \leq i \leq k) \lambda'_i \geq \lambda_{i_1} \geq 0$ .

## C Proof of Theorem 2

The proof requires the next intermediary results.

**Theorem 6.** *Consider  $K$  a compact subset of  $\mathbb{R}^n$  and  $f$  a continuous application,  $f(K)$  is a compact set.*

**Lemma 3.** *Let  $E = \{\mathbf{x}_1, \dots, \mathbf{x}_k\}$  be a finite subset of  $\mathbb{R}^n$ . The convex hull  $\text{Conv}(E)$  of  $E$  is a closed set.*

*Proof.* Let

$$K = \{(\lambda_1, \dots, \lambda_k) \mid \forall i (\lambda_i \in [0, 1]) \wedge \sum_{i=1}^k \lambda_i = 1\} \subset \mathbb{R}^k$$

and

$$f : (\lambda_1, \dots, \lambda_k) \mapsto \sum_{i=1}^k \lambda_i \mathbf{x}_i$$

Since  $K$  is a compact set and  $f$  is a continuous application,  $f(K)$  is a closed set. We conclude by noticing that  $\text{Conv}(E) = \text{Conv}(\{\mathbf{x}_1, \dots, \mathbf{x}_k\}) = f(K)$ .

We now prove Theorem 2.

*Proof.* We prove the two directions of the equivalence.

- We have  $TC(Conv_{2^*}(E)) \subset Conv(E)$ . Indeed, we have  $TC(Conv_{2^*}(E)) \subset TC(Conv(E)) \subset Conv(E)$ . The first inclusion holds because  $Conv_{2^*}(E) \subset Conv(E)$  and, for any  $E_1, E_2 \in \mathbb{R}^n$ ,  $E_1 \subset E_2 \Rightarrow TC(E_1) \subset TC(E_2)$ . The second inclusion holds because of Lemma 3.
- By Lemma 1, we have  $Conv(E) \subset TC(Conv_{2^*}(E))$ .

## D Proof of Theorem 3

We first state and prove a series of theorems and lemmas. We denote by  $W^{-1}(w, n)$  the unique vector  $\mathbf{v}$  of  $\mathbb{R}^n$  such that  $w \in W(\mathbf{v}, n)$ .

**Lemma 4.** *Let  $A$  be a RVA that represents a set  $E \subset \mathbb{R}^n$ . Let  $A'$  be  $A$  with all states of its fractional part made accepting. We have*

$$\forall(w' \in L(A')) \forall(k \in \mathbb{N}) \exists(w \in L(A)) (pref_k(w) = pref_k(w')).$$

**Lemma 5.** *Consider  $E \subset \mathbb{R}^n$  and  $\mathbf{v} \in \mathbb{R}^n$ . If for each  $\epsilon > 0$  there exists  $\mathbf{v}' \in E$  such that  $|\mathbf{v} - \mathbf{v}'| \leq \epsilon$ , then there exists  $w \in W(\mathbf{v}, n)$  such that*

$$\forall(k \in \mathbb{N}) \exists(\mathbf{v}' \in E, w' \in W(\mathbf{v}', n)) (pref_k(w) = pref_k(w')).$$

**Lemma 6.** *Let  $A$  be a RVA that represents a set  $E \subset \mathbb{R}^n$ . Let  $A'$  be  $A$  with all states of its fractional part made accepting. Consider  $w$ , an encoding of a vector  $\mathbf{v}$  of  $\mathbb{R}^n$ . If for each  $k \in \mathbb{N}$  there exists  $w' \in L(A)$  such that  $pref_k(w) = pref_k(w')$ , then  $w \in L(A')$ .*

**Theorem 7.** *Let  $A$  be a RVA that represents a set  $E \subset \mathbb{R}^n$ . Let  $A'$  be  $A$  with all states of its fractional part made accepting. For each  $w' \in L(A')$ , the vector  $W^{-1}(w', n)$  is in  $TC(E)$ .*

*Proof.* The result is obvious if  $w' \in L(A)$ . Assume now that  $w' \notin L(A)$ . By Lemma 4, we have

$$\forall(w' \in L(A')) \forall(k \in \mathbb{N}) \exists(w \in L(A)) (pref_k(w) = pref_k(w')).$$

Which implies

$$\forall(\mathbf{v}' \in W^{-1}(L(A'), n)) \forall(\epsilon > 0) \exists(\mathbf{v} \in W^{-1}(L(A), n)) (|\mathbf{v} - \mathbf{v}'| \leq \epsilon).$$

This concludes the result.

**Theorem 8.** *Let  $A$  be a RVA that represents a set  $E \subset \mathbb{R}^n$ . Let  $A'$  be  $A$  with all states of its fractional part made accepting. For each  $\mathbf{v}$  of  $TC(E)$ , there exists  $w \in W(\mathbf{v}, n)$  such that  $w \in L(A')$ .*

*Proof.* Let  $\mathbf{v}$  be a vector of  $\mathbb{R}^n$ . If  $\mathbf{v} \in E$ , then the result is obvious. Consider now that  $\mathbf{v}$  does not belong to  $E$ .

By definition of the topological closure, we have

$$\forall(\epsilon > 0)\exists(\mathbf{v}' \in E) (|\mathbf{v} - \mathbf{v}'| \leq \epsilon).$$

By Lemma 5, there exists  $w \in W(\mathbf{v}, n)$  such that

$$\forall(k \in \mathbb{N})\exists(\mathbf{v}' \in E, w' \in W(\mathbf{v}', n)) (pref_k(w) = pref_k(w')).$$

Using Lemma 6, we conclude that  $w \in L(A')$ .

We now observe that Theorem 3 is a direct consequence of the two theorems above.

## E Preciseness

This section contains the full version of Section 5.3.

Checking preciseness could be performed with the techniques proposed in [6]. However, this solution is computationally demanding and not really practical. In the present situation, one can propose a more efficient scheme that exploits the properties of the extrapolation. We first introduce the following definition and theorem.

**Definition 8.** *Let  $E \subseteq \mathbb{R}^n$  be a convex set. The set of extreme points of  $E$ , denoted  $S(E)$ , is defined as  $\{\mathbf{x} \in E \mid (\neg\exists(\mathbf{x}_1, \mathbf{x}_2) \in E)(\mathbf{x}_1 \neq \mathbf{x}_2 \wedge \mathbf{x} = (\mathbf{x}_1 + \mathbf{x}_2)/2)\}$ .*

By extension we will also consider that the notation  $S(E)$  extends on automata representing vector sets

**Theorem 9.** *(Krein-Milman) Let  $E \subseteq \mathbb{R}^n$  be a compact convex set. The set  $E$  is the convex hull of its set of extreme points.*

We observe that if the extrapolation of the limit of the median sequence of a set is safe, then its topological closure is a compact convex set.

**Lemma 7.** *Let  $TC(A_E^*)$  be the RVA that represents the topological closure of a safe extrapolation of the limit of the median sequence of a finite set  $E \subseteq \mathbb{Z}^n$ . The set represented by  $TC(A_E^*)$  is a compact convex set.*

*Proof.* The fact that  $TC(A_E^*)$  is convex is a direct consequence of applying Theorem 4 to  $TC(A_E^*)$  rather than to  $A_E^*$ . The set  $TC(A_E^*)$  is closed by construction. Finally, the fact that  $TC(A_E^*)$  is bounded follows from the fact that the extrapolation step only modifies the fractional part of the RVA.

We then have the following theorem.

**Lemma 8.** *Let  $A_E^*$  be a RVA that represents a safe extrapolation of the limit of the median sequence of a finite set of integer vectors represented by the RVA  $A_E$ . If  $L(S(TC(A_E^*))) \subseteq L(A_E)$ , then  $L(TC(A_E^*)) \subseteq L(\text{Conv}(A_E))$ .*

*Proof.* If  $L(S(TC(A_E^*))) \subseteq L(A_E)$ , then  $L(\text{Conv}(S(TC(A_E^*)))) \subseteq L(\text{Conv}(A_E))$ . By Lemma 7, we have that  $TC(A_E^*)$  is a compact convex set. We can apply Krein-Milman's theorem and obtain that  $L(TC(A_E^*)) = L(\text{Conv}(S(TC(A_E^*)))) \subseteq L(\text{Conv}(A_E))$ .

In summary, to check the preciseness of a RVA  $A_E^*$  that represents a safe extrapolation of the limit of the median sequence of a finite set  $E \subseteq \mathbb{Z}^n$ , we first compute a RVA  $TC(A_E^*)$  for the topological closure of the set represented by  $A_E^*$ . We then compute an automaton for  $S(TC(A_E^*))$ , which is easily done by computing the difference between  $TC(A_E^*)$  and  $C_2(TC(A_E^*))$ . Finally, one checks whether the language of the resulting automaton is included in the one of  $A_E$ . Again, all complementation operations are only applied to weak deterministic Büchi automata.

## F Experiments

We report experiments for both finite and infinite sets of integers.

### F.1 For finite sets

In Table 1 we give the vertices of some of the convex sets that were considered. We also give the number of states of the RVA that represents those sets, of the RVA that represents the largest element in the median sequence, and of the RVA that represents the convex hull. The same information is given for the difference/union of finite convex sets in Table 2 and for arbitrary finite sets of points in Table 3. All those examples were handled in less than a minute. We also tested our technique on sets of higher dimensions. The efficiency of the technique decreases when the dimension of the set increases. This is not surprising since computing the elements of the median sequence of a set over  $\mathbb{R}^n$  requires to compute and determinize RVAs representing sets over  $\mathbb{R}^{2n+1}$ .

### F.2 For infinite sets

Table 4, presents some of the results we obtained by applying our technique to infinite sets. We compared those results with a directly computed RVA representing the convex hull of the initial set, and observed that they coincide except for the set of Example 2. This means that we did not encounter situations where T(O)RMC produced a safe but not precise extrapolation.

Finite convex in $\mathbb{Z}^n$			
Vertices	$ A_E $	$ A_E^i $	$ A_{Conv(E)} $
(1), (2)	7	9	7
(-1,7), (5,-6)	28	290	104
(-13,1), (11,0)	40	354	142
(0,2), (0,4), (2,6), (4,4), (4,2)	54	78	58
(0,0,0), (3,3,2)	63	110	100
(1,1,1), (3,3,2), (2,2,4)	86	286	127
(-1,0,-1), (-1,2,-1), (0,1,-1), (0,1,1)	72	205	97

**Table 1.** Convex hull for finite convex sets.

Non Convex in $\mathbb{Z}^n$			
Description	$ A_E $	$ A_E^i $	$ A_{Conv(E)} $
$[(0,0), (4,4), (8,0)] \setminus [(4,0), (4,2), (6,0)]$	65	97	61
$[(0,0), (3,3), (6,3), (6,0)] \cup [(6,0), (6,3), (9,6), (9,0)]$	62	174	73
$[(0,0,0), (0,2,0), (0,2,2), (3,0,0), (3,2,0), (3,2,2)] \cup [(0,0,0), (0,2,0), (0,0,2), (3,0,0), (3,2,0), (3,0,2)]$	170	283	160
$[(-1,0,-1), (-1,2,-1), (0,1,-1), (0,1,1)] \cup [(-1,0,3), (-1,2,3), (0,1,3), (0,1,1)]$	96	337	134
$[(0,0,0), (0,3,0), (3,0,0), (3,3,0), (0,0,5), (0,3,5), (3,0,5), (3,3,5)] \setminus [(1,1,0), (1,2,0), (2,1,0), (2,2,0), (1,1,5), (1,2,5), (2,2,5), (2,1,5)]$	218	265	184
$[(0,3,0), (0,4,0), (3,3,0), (3,4,0), (0,0,3), (3,0,3), (3,7,3), (0,7,3)] \setminus [(1,0,1), (1,0,2), (2,0,2), (2,0,1), (1,7,1), (2,7,1), (1,7,2), (2,7,2)]$	227	334	219

**Table 2.** Convex hull for the difference/union between finite convex sets.

Sets of points			
Points of the set	$ A_E $	$ A_E^i $	$ A_{Conv(E)} $
(0,0), (6,3)	27	97	39
(0,0), (3,3), (4,3)	31	314	61
(0,0), (3,3), (6,3), (9,6), (9,0)	42	686	73
(1,1,1), (3,2,1), (2,2,4)	64	370	137
(0,0,0), (0,2,0), (0,0,2), (0,2,2), (0,1,1), (3,0,0), (3,2,0), (3,0,2), (3,1,1), (3,2,2)	126	556	160

**Table 3.** Convex hull for finite sets of points.

Infinite sets			
Description	$ A_E $	$ A_E^i $	$ A_{Conv(E)} $
$(y = x \cup y = -x) \cap y \geq 0$	23	44	25
$(y = x + 1 \cup \{(0, 0)\}) \cap y \geq 0$	29	69	28
$x \geq 1 \cap x \leq 2 \cap y \geq 0 \cap y \leq 1 \cap z \geq 1$	79	133	78
$(-2x + z \leq 1 \cap x - y \leq -1 \cap x + y \leq 1) \cup$ $(-2x - z \leq -1 \cap x - y \leq -1 \cap x + y \leq 1)$	159	159	44
$-2x + z \leq 1 \cap x - y \leq -1 \cap x + y \leq 1$	114	180	119
$(x - y \leq 0 \cap -x - y \leq 0) \cup$ $(-x + y \leq 0 \cap x + y \leq 0)$	36	36	4

**Table 4.** Convex hull for infinite sets.