International Journal of Foundations of Computer Science

Computing Convex Hulls by Automata Iteration*

François Cantin†
Institut Montefiore
Université de Liège, 4000 Liège, Belgium
cantin@montefiore.ulg.ac.be

Axel Legay‡
Computer Science Department,
Carnegie Mellon University, Pittsburgh, PA,
alegay@cs.cmu.edu

Pierre Wolper§
Institut Montefiore
Université de Liège, 4000 Liège, Belgium
pw@montefiore.ulg.ac.be

Received (Day Month Year)
Accepted (Day Month Year)
Communicated by (xxxxxxxxxx)

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 implemented.

Keywords: Convex Hull, Büchi Automata, (ω-)Regular Model Checking

1. Introduction

Automata-based representations for sets of integer and real vectors have been a subject of growing interest in recent years [1, 3, 12, 21, 24]. While usually not op-

*A preliminary version of this paper appeared in the Proceedings of the 13th International Conference on Implementation and Application of Automata.
†This author is supported by a F.R.I.A grant.
‡This author was supported by a F.R.I.A. grant, and by the EU network of excellence ARTIST 2. He is now supported by a B.A.E.F grant.
§This author is supported by the FRFC project "centre fédéré en vérification" funded by the Belgian National Science Fundation (FNRS) under grant nr. 2.4530.02.

timal 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 [4]. 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 [9, 2, 7], but going in the other direction is substantially more difficult. Nevertheless, it has been shown that it is possible [22] 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 of the automata to formulas transformation 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, the automaton lacks the structure needed to construct the constraints defining the represented set [18, 22]. 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, 22] could then be applied to obtain constraints[a].

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 the real numbers) 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. Of course even when being interpreted over the integers, the constraints defining the real convex hull of a non convex set of integer vectors will over approximate this set. However, in many applications of automata-based representations such as model checking (see [14] for an example), this over approximation is known to be of interest and should not be seen as a drawbacks of our approach.

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 [5]. 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

[a]Developing such techniques is an open problem.

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 [5], which have so far only been applied in the context of "regular model checking" [8], 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 [5], 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 [5] is probably not practical (see [19]), 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 is to establish the basis of a different approach with interesting characteristics, performance gains are not 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. Finally, we mention that our approach extends to infinite sets under some restrictions.

Related Work 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 one finds a long series of algorithms specialized to the 2D and 3D case and widely used and studied in computational geometry. Algorithms for the general case (any dimensions) have also been studied [11]. All those algorithms, which are generally more efficient than an automata-based approach, require an enumeration of the set, which we avoid here.

In [13], Finkel and Leroux show that the convex hull of a (possibly infinite) set of integer vectors represented by a regular expression (and hence also an automaton) is a computable polyhedron. The paper gives a concise existence proof of an algorithm, but does not discuss the applicability of the method and implementation issues. Also note that the given algorithm computes the topological closure of the convex hull, hence side stepping one of the problems of dealing with infinite sets we discuss in Section 6. Finally, the algorithm goes from a regular expression to a polyhedron represented by rays, which does not conform to our goal of staying within automata-based representations. Thus, while from a theoretical point of view this result is in

some ways more general than what is presented in this paper, it is quite orthogonal to our goal of exploring applications of automata based computations on arithmetic sets.

In very recent work [23], Leroux showed that the closed convex hull of a set of reals that can be represented by an infinite-word automaton is rational polyhedral. In [23], Leroux also proposed an algorithm to compute the constraints defining this convex hull. This seems to be quite a promising approach, though notice that it is again the closure of the convex hull that is computed. The approach works by reducing the problem to a data-flow analysis problem to be solved on the automaton graph. Solving this data-flow analysis problem has some similarity with our automata sequence extrapolation step and exploring the link between the two would certainly be interesting future work.

2. Automata-theoretic background

2.1. *Automata on infinite words*

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

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, Σ is the input alphabet, $\delta : Q \times \Sigma \to 2^Q$ is a *transition function* ($\delta : Q \times \Sigma \to Q$ if the automaton is deterministic), q_0 is the initial state, and F is a set of accepting states. A *run* π of a Büchi automaton $A = (Q, \Sigma, \delta, q_0, F)$ on an ω-word w is a mapping $\pi : \mathbb{N} \to Q$ 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 $inf(\pi)$ be the set of states that occur infinitely often in a run π. A run π is said to be accepting if $inf(\pi) \cap F \neq \emptyset$. An ω-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 ω-words it accepts.

A *co-Büchi automaton* is defined exactly as a Büchi automaton except that its accepting runs are those for which $inf(\pi) \cap F = \emptyset$.

We will also use the notion of *weak* automata [28]. For a Büchi automaton $A = (Q, \Sigma, \delta, q_0, F)$ to be *weak*, there has to be a partition of its state set Q into disjoint subsets $Q_1, \ldots, Q_m$ such that for each of the Q_i either $Q_i \subseteq F$ or $Q_i \cap F = \emptyset$; and there is a partial order $\leq$ on the sets $Q_1, \ldots, Q_m$ such that for every $q \in Q_i$ and $q' \in Q_j$ for which, for some $a \in \Sigma$, $q' \in \delta(q, a)$ ($q' = \delta(q, a)$ in the deterministic case), $Q_j \leq Q_i$. Roughly speaking, a weak automaton is thus 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 [28]. However, there are algorithmic advantages to working with weak automata. Indeed, weak determinis-

tic 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 [30]. There exists an easy to implement determinization procedure for weak automata [27, 16], which produces Büchi automata that are deterministic, but not necessarily weak. The procedure is exponential in the size of the automaton.

Nevertheless, if the represented language can be accepted by a weak deterministic automaton, the result of the determinization procedure will be *inherently weak* according to the definition of [4] and thus easily transformed into a weak automaton.

Definition 1. *A Büchi automaton is* inherently weak *if none of the reachable strongly connected components of its transition graph contain both accepting (visiting at least one accepting state) and non-accepting (not visiting any accepting state) cycles.*

This property yields a pragmatic approach for staying, when at all possible, within the realm of deterministic weak Büchi automata. Indeed, we start with sets represented by such automata and being weak deterministic is preserved by union, intersection, synchronous product, and complementation. If a projection is needed, the result is determinized by the procedure proposed in [27, 16]. Then, either the result is inherently weak and we can proceed, or it is not and we are then confronted to a set that cannot be represented by a weak deterministic automaton.
Finally, a major advantage of weak deterministic Büchi automata is that they admit a minimal form, which is unique up to isomorphism [25].

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 [7].

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 $\{0, \ldots, r-1\}$, the special symbol "$\star$" is a separator, and w_F encodes the fractional part x_F of x as an infinite word over $\{0, \ldots, r-1\}$. 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 $\{0, \ldots, r-1, \star\}$, or as a single word w over the alphabet $\{0, \ldots, r-1\}^n \cup \{\star\}$.

Example 2. *In base* 2*, the vector* $(-2, 12.3)$ *can be encoded as*

$$(11110 \star 0^\omega, 01100 \star 01[1001]^\omega)$$

or as the word

$$(1,0)(1,1)(1,1)(1,0)(0,0) \star (0,0)(0,1)[(0,1)(0,0)(0,0)(0,1)]^\omega.$$

Using an alphabet of size r^n is clearly going to be problematic as soon as n starts to grow. The solution proposed in [4, 31] 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 size of the base r. This scheme is referred as the *serial encoding* as opposed to the *simultaneous encoding* in which the alphabet consists of tuples of digits.

Example 3. *Using the serial encoding, the vector* $(-2, 12.3)$ *can be encoded in base* 2 *as*

$$1011111000 \star 0001[01000001]^\omega.$$

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. We use $W^{-1}(w, n)$ to denote the unique vector $\mathbf{v} \in \mathbb{R}^n$ such that $w \in W(\mathbf{v}, n)$.

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 4. *Let* $n > 0$ *and* $r > 1$ *be integers. A base-*r *n-dimension* Real Vector Automaton (RVA) *is a Büchi automaton* $\mathcal{A}$ *automaton such that*

- *Every word accepted by* $\mathcal{A}$ *is an encoding in base* r *of a vector in* $\mathbb{R}^n$*, and*
- *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.*

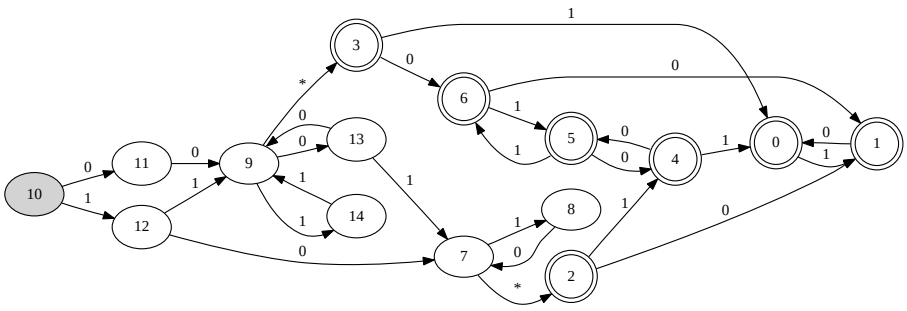

Fig. 1. An RVA for $x_2 = x_1 + 1/2$.

A RVA is said to *represent* the set of vectors encoded by the words that belong to its accepted language. The set of *fractional* states of a RVA A, denoted by Q_F^A, is the subset of Q that contains all the states of A that can be reached after having followed a transition labeled by $\star$.

The expressive power of RVAs has been studied in [6] and corresponds exactly to linear arithmetic over the reals and integers, i.e., $\langle \mathbb{R}, +, \leq, Z \rangle$ (where Z is a predicate that is true if and only if its argument is an integer), extended with a special base-dependent predicate V_r that can check the value of the digit appearing in a given position. Concretely, given a base r, V_r enrichs $\langle \mathbb{R}, +, \leq, Z \rangle$ with the multiplication by a power of r. If V_r is not used or if the sets represented are finite, RVAs can always be constructed to be weak deterministic automata [4]. In other situations, we have to check whether the resulting automaton is inherently weak. Using the algorithms described in [4], a RVA that represents a finite set can always be constructed to be a weak deterministic automaton. 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 (see Section 6.2). Finally, it is worth mentioning that operations on sets (union, intersection, Cartesian product, complementation, projection) directly extend to operations on RVA [7] (example : The RVA that represented the union of two sets represented by two RVAs is the union of the two automata).

Example 5. *The Büchi automaton in Figure 1 is a serialized RVA representing all the encodings in base* 2 *of vectors that are solution to the equation* $x_2 = x_1 + 1/2$. *The initial state of the automaton is colored in gray and the accepting states are surrounded by a double circle. The set of fractional states of the automaton is given by* $\{0, 1, 2, 3, 4, 5, 6\}$.

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 integer, rational, and real numbers, respectively. 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 *ith* 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* if and only if 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 6. *Given a set $E \subseteq \mathbb{R}^n$, the* convex hull *of E is the set $Conv(E) \subseteq \mathbb{R}^n$ defined by*

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

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 (contained in a ball of finite radius) and closed set. We use the concept of *topological closure* of a set.

Definition 7. *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

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

where $common(w, w')$ 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 $E = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_k\}$ 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 8. *The median sequence of E is the* infinite *sequence $E_0, E_1, E_2, \ldots$ 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 $Conv(E)$. However, E^* is not the complete convex hull, but can be characterized using the following definition.

Definition 9. *The 2-chopped convex hull of a finite subset $E = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_k\}$ of $\mathbb{Z}^n$ is the* maximal *subset $Conv_{2^*}(E)$ of $Conv(E)$, where for each $\mathbf{v} \in Conv_{2^*}(E)$, $\mathbf{v} = \sum_{i=1}^{k} \lambda_i \mathbf{x}_i$ 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 $i \in [1, \ldots, k]$.*

Theorem 10. *For any finite subset $E = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_k\}$ of $\mathbb{Z}^n$, the limit of its median sequence and its 2-chopped convex hull coincide, i.e $E^* = Conv_{2^*}(E)$.*

Proof. 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 both a 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 first prove $E^* \subset Conv_{2^*}(E)$. By construction, each vector $\mathbf{v} \in E^*$ can be expressed as a 2-term of E. Moreover, it is easily proved by induction on its depth that 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 $\forall(1 \leq i \leq k)\,[(0 \leq a_i \leq 1) \wedge (\exists(k_i, m_i \in \mathbb{N})\,(a_i = \frac{k_i}{2^{m_i}}))]$ and $\sum_{i=1}^{k} a_i = 1$.

We now prove $Conv_{2^*}(E) \subset E^*$. Similarly, it is not difficult to see that each vector of $Conv_{2^*}(E)$ 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 $j \geq i$. □

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. This is proved in Lemma 12, which is based on the following result.

Lemma 11. *Consider $E = \{\mathbf{x}_1, \ldots, \mathbf{x}_k\}$, 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 $\forall(1 \leq i \leq k)$ $(|\lambda_i - \lambda'_i| \leq \epsilon_i)$ with $\epsilon_i > 0$ and such that $\sum_{i=1}^{k} \epsilon_i \leq \frac{\epsilon}{\sqrt{n} x_{max}}$, then $|\mathbf{v} - \mathbf{v}'| \leq \epsilon$.*

Proof. We have the following development.

$$(\forall(1 \leq i \leq k)\,\exists(\epsilon_i > 0)\,(|\lambda_i - \lambda'_i| \leq \epsilon_i)) \wedge (\sum_{i=1}^{k} \epsilon_i \leq \frac{\epsilon}{\sqrt{n}\, x_{max}}))$$

$$\Leftrightarrow \sum_{i=1}^{k} |\lambda_i - \lambda'_i| \leq \frac{\epsilon}{\sqrt{n}\, x_{max}}$$

$$\Leftrightarrow \sqrt{\left(\sum_{i=1}^{k} |\lambda_i - \lambda'_i|\right)^2} \leq \frac{\epsilon}{\sqrt{n}\, x_{max}}$$

$$\Leftrightarrow \sqrt{n}\, x_{max} \sqrt{(|\lambda_1 - \lambda'_1| + \cdots + |\lambda_k - \lambda'_k|)^2} \leq \epsilon$$

$$\Leftrightarrow \sqrt{n\, x_{max}^2\, (|\lambda_1 - \lambda'_1| + \cdots + |\lambda_k - \lambda'_k|)^2} \leq \epsilon$$

$$\Leftrightarrow \sqrt{n\, (|\lambda_1 - \lambda'_1|\, x_{max} + \cdots + |\lambda_k - \lambda'_k|\, x_{max})^2} \leq \epsilon$$

$$\Leftrightarrow \ [\ (|\lambda_1 - \lambda'_1|\, x_{max} + \cdots + |\lambda_k - \lambda'_k|\, x_{max})^2 + \cdots$$
$$+\ (|\lambda_1 - \lambda'_1|\, x_{max} + \cdots + |\lambda_k - \lambda'_k|\, x_{max})^2\ ]^{\frac{1}{2}} \leq \epsilon. \qquad (1)$$

By Minkowski inequality [26], for all $1 \leq i \leq n$, we have

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

Therefore, from (1) we deduce that

$$[\ ((\lambda_1 - \lambda'_1)\, \mathbf{x}_1[1] + \cdots + (\lambda_k - \lambda'_k)\, \mathbf{x}_k[1])^2 + \cdots$$
$$+\ ((\lambda_1 - \lambda'_1)\, \mathbf{x}_1[n] + \cdots + (\lambda_k - \lambda'_k)\, \mathbf{x}_k[n])^2\ ]^{\frac{1}{2}} \leq \epsilon$$

$$\Leftrightarrow \ [\ ((\lambda_1\mathbf{x}_1[1] + \cdots + \lambda_k\mathbf{x}_k[1]) - (\lambda'_1\mathbf{x}_1[1] + \cdots + \lambda'_k\mathbf{x}_k[1]))^2 + \cdots$$
$$+\ ((\lambda_1\mathbf{x}_1[n] + \cdots + \lambda_k\mathbf{x}_k[n]) - (\lambda'_1\mathbf{x}_1[n] + \cdots + \lambda'_k\mathbf{x}_k[n]))^2\ ]^{\frac{1}{2}} \leq \epsilon$$

$$\Leftrightarrow \sqrt{(\mathbf{v}[1] - \mathbf{v}[1]')^2 + \cdots + (\mathbf{v}[n] - \mathbf{v}[n]')^2} \leq \epsilon$$
$$\Leftrightarrow |\mathbf{v} - \mathbf{v}'| \leq \epsilon. \qquad \square$$

Lemma 12. *For each* $\mathbf{v} \in Conv(E)$ *and* $\epsilon > 0$*, there exists* $\mathbf{v}' \in Conv_{2^*}(E)$ *such that* $|\mathbf{v} - \mathbf{v}'| \leq \epsilon$.

Proof. We define $x_{max} = max_{i,j}(|\mathbf{x}_i[j]|)$, with $i \in [1, k]$ and $j \in [1, n]$. For each $\mathbf{v} \in Conv(E)$ and each $\epsilon > 0$, we build a vector $\mathbf{v}' \in Conv_{2^*}(E)$ with $|\mathbf{v} - \mathbf{v}'| \leq \epsilon$. This amounts to assign a value to each λ'_i. This assignation is direct if $\mathbf{v} \in Conv_{2^*}(E)$. Assume now that $\mathbf{v} \notin Conv_{2^*}(E)$. By hypothesis, we have

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

By Lemma 11, 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 $|\mathbf{v} - \mathbf{v}'| \leq \epsilon$.

Assume $l \in \mathbb{N}$ such that $k\, 2^{-l} \leq \frac{\epsilon}{\sqrt{n} x_{max}}$. For each $1 \leq i \leq k$, we define λ_{i_1} by truncating the binary encoding of λ_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}.

Let $d = \frac{\sum_{i=1}^{k} \lambda_{i_2}}{2^{-l}}$. 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 λ'_i as follows:

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

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 11, $|\mathbf{v} - \mathbf{v}'| \leq \epsilon$.

To conclude, observe that we have the following.

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

From Lemma 12 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 13. *For any finite subset $E = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_k\}$ of $\mathbb{Z}^n$, we have that $TC(Conv_{2^*}(E)) = Conv(E)$.*

Proof. By Lemma 12, we have that $Conv(E) \subseteq TC(Conv_{2^*}(E))$. We can also show that $TC(Conv_{2^*}(E)) \subseteq Conv(E)$. Indeed, we have $TC(Conv_{2^*}(E)) \subseteq TC(Conv(E)) \subseteq Conv(E)$. The first inclusion holds because $Conv_{2^*}(E) \subseteq Conv(E)$ and, for any $E_1, E_2 \in \mathbb{R}^n$, $E_1 \subseteq E_2 \Rightarrow TC(E_1) \subseteq TC(E_2)$. The second inclusion holds because the convex hull of a finite subset of $\mathbb{R}^n$ is always a closed set. □

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 $Conv_{2^*}(E)$ and $TC(E)$ for a set E represented by a *RVA*.

5. Algorithmic issues

We consider a finite subset $E = \{\mathbf{x_1}, \mathbf{x_2}, \ldots, \mathbf{x_k}\}$ of $\mathbb{Z}^n$ that is represented by a (weak deterministic) RVA A_E. Our goal is to compute a 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 a RVA A_{E^*} representing the limit E^* of the median sequence of E, and then computing a RVA representing the topological closure of E^*. We now show how these two problems can be tackled by automata-based algorithms. In the rest of this section, the RVA that represents the i-th element of the median sequence of E is denoted by A^i_E.

5.1. *Computing a RVA for the 2-chopped Hull*

5.1.1. *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 an automaton in this class. Indeed, from Definition 8 it is easy to see that computing a RVA for the set E_{i+1} from a RVA that represents the set E_i can be done with the following operations: Cartesian product, projection, union, and intersection. As discussed in Section 2 and in [4] these operations keep us within the sets representable by weak deterministic RVA.

5.1.2. *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 [5] and detailed in [19]. 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 nonaccepting strongly connected component 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 [29, 20]. The tool relies on the LASH package [17] for automata manipulation procedures, but implements the specific algorithms given in [5].

It is worth mentioning that the automata produced by T(O)RMC are weak, but not necessarily deterministic [5]. Furthermore, if one tries to determinize these automata, one might end up combining accepting and non accepting connected components[b], 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 [4].

[b]This can only occurs because the extrapolation procedure adds non accepting strongly connected components that correspond to the repetition of the increment.

We conclude the section with the following observations.

- An extrapolation makes sense in the context of the convex hull computation, only if the increments are detected among the fractional states of the automaton. Indeed, an increment in the fractional part leads to a denser set, as required, whereas an increment in the integer part would lead to adding an unlimited number of new integer values, which cannot be needed for computing the convex hull of a finite set. Thus, we will only allow T(O)RMC to extrapolate within the fractional part of the automaton.
- 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. An interesting open question is whether termination can be guaranteed in the specific case of the convex hull computation we are considering.
- As discussed in [19], the operations performed for computing an extrapolation from a finite sequence of automata can be done in time linear in the size of the sets of states of the automata in the sequence.

5.2. *Computing the topological closure of a RVA-represented set*

In this section, we explicitely consider RVAs that may not be weak deterministic. Consider a set $E \subseteq \mathbb{R}^n$ represented by a RVA A_E. Our goal is to compute a 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 is shown by the following result.

Theorem 14. *Let A_E be a RVA representing a vector set E. Let $\overline{A}_E$ be A_E with all states of its fractional part from which an accepting state is reachable 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 $\mathbf{v} \in TC(E)$.*

Proof. We prove the two directions of the equivalence.

(1) We first show that each word that belongs to the language of $\overline{A}_E$ is the encoding of a vector that is in the topological closure of E. Indeed, it is easy to see that

$$\forall(w \in L(\overline{A}_E))\forall(k \in \mathbb{N})\exists(w' \in L(A_E))(pref_k(w) = pref_k(w')),$$

which implies the definition of the topological closure

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

(2) We now show that for each vector $\mathbf{v} \in \mathbb{R}^n$, if $\mathbf{v} \in TC(E)$, then $W(\mathbf{v}, n) \cap L(\overline{A}_E) \neq \emptyset$.
By definition of the topological closure, we have

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

It is easy to see that there exists $w \in W(\mathbf{v}, n)$ such that

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

and we can thus conclude that $w \in L(\overline{A}_E)$. □

Theorem 14 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. This is illustrated with the following example.

Example 15. *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 set $0^*01000\star(01)^\omega$ that corresponds to the high encoding of 2 and the low encoding of 1 is never added.*

For each vector $\mathbf{v} \in TC(E)$, $\overline{A}_E$ thus contains at least one combination of the encodings of each component. In fact, since the abstraction always occurs in the fractional part, it is easy to see that all other combinations that can be obtained by increasing/decreasing the length of the integer-part of the encoding of each component of $\mathbf{v}$ are also included. In this context, missing combinations can easily be added by automata-based operations [10], including projection after which an exponential determinization step is needed.

5.3. *Correctness criterion*

After having constructed the extrapolation A^*_E of a finite sequence $A^{i_1}_E$, $A^{i_2}_E$, …, $A^{i_l}_E$ 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 a RVA A, $Conv(A)$ is an RVA that represents the convex hull of the set represented by A.

5.3.1. *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 16. *Let A^*_E and A_{E^*} be the RVAs that represent the extrapolation of a median automata sequence for a set E and the actual limit of this sequence, respectively. Assume that E is represented by the RVA A_E. We have that, if $L(C_2(A^*_E)) \subseteq L(A^*_E)$, then $L(A_{E^*}) \subseteq L(A^*_E)$.*

Proof. Recall that $L(A_{E^*}) = \bigcup_i L(A^i_E)$, where A^i_E is the RVA representing the i-th elment in the median sequence of origin E. We show that for each i, $L(A^i_E) \subseteq L(A^*_E)$. The base case, i.e., $L(A^0_E) \subseteq L(A^*_E)$, holds by hypothesis. Suppose now that $i > 0$ and that the result holds for any $k < i$. We have that $L(A^i_E) \subseteq L(C_2(A^{i-1}_E)) \subseteq L(C_2(A^*_E)) \subseteq L(A^*_E)$. □

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 Theorem 16. We have the following result.

Theorem 17. *Let A^*_E and A_{E^*} be the RVAs that represent the extrapolation of a median automata sequence for a set E and the actual limit of this sequence, respectively. If $L(C_2(TC(A^*_E))) \subseteq L(TC(A^*_E))$, then $L(TC(A_{E^*})) = L(Conv(A_E)) \subseteq L(TC(A^*_E))$.*

The advantage of computing the topological closure before performing the safety check is that the strongly connected components added by T(O)RM are made uniformly accepting by the procedure that computes the topological closure. This ensures that we only need to complement weak deterministic automata.

5.3.2. *Preciseness*

Checking preciseness could be performed with the techniques proposed in [5, 19]. However, this solution which involves complex data-structures is computationally demanding and not really practical [19]. In the present situation, one can however propose a much more efficient scheme that exploits the properties of the extrapolation. We use the following definition.

Definition 18. *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 use the notation $S(A)$ on automata representing vector sets. We will also use the following theorem.

Theorem 19. *(Krein-Milman [15]) 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 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. We first 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 20. *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 Theorem 17. 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.

Theorem 21. *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 $L(TC(A^*_E)) \subseteq L(Conv(A_E))$.*

Proof. If $L(S(TC(A^*_E))) \subseteq L(A_E)$, then $L(Conv(S(TC(A^*_E)))) \subseteq L(Conv(A_E))$. By Lemma 20, we have that $TC(A^*_E)$ is a compact convex set. We can apply Krein-Milman's theorem and obtain that $L(TC(A^*_E)) = L(Conv(S(TC(A^*_E)))) \subseteq L(Conv(A_E))$. □

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. As for the safety case, all complementation operations are only applied to weak deterministic Büchi automata.

6. Observations

6.1. *Infinite Sets*

It is worth mentioning that our results do not extend 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 22. *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 $Conv(E) = \{(x, y) \in \mathbb{R}^2 \mid$*

$(y \leq x+1) \wedge (y \geq 0) \wedge (y > x)\}$, *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* $Conv(E)$.

Observe also that, in the present context, we cannot check for 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.

As a conclusion, when working with infinite sets, the best we can propose is a convex overapproximation of the real convex hull.

6.2. *Other Bases*

We have already mentioned that the definition of the median sequence and the encoding of real numbers in base 2 are linked. We have noticed that when working in a base $r > 2$ T(O)RMC was not able to detect increments. This may be explained as follows. We can observe that in base 2, any word in the language of an automaton that represents an element in the median sequence can be obtained by adding an increment[c] to one of the accepting words of the RVA representing the previous element in the sequence. However, when applying the construction given in Definition 8 in a base $r > 2$, we cannot make the same observation. This may explain why increments do not appear between the automata of the sequence. The solution is to generalize the construction of the successive elements of the median sequence in such a way that the base is taken into account. For this we propose to use the following computation.

$$\begin{aligned} E_{i+1} = E_i \cup \{(\mathbf{x_1} + (r-1)\mathbf{x_2})/r \mid \mathbf{x_1}, \mathbf{x_2} \in E_i\} \\ \cup \{(2\mathbf{x_1} + (r-2)\mathbf{x_2})/r \mid \mathbf{x_1}, \mathbf{x_2} \in E_i\} \\ \cup \ldots \cup \{((r-1)\mathbf{x_1} + \mathbf{x_2})/r \mid \mathbf{x_1}, \mathbf{x_2} \in E_i\} \end{aligned}$$

One can then easily adapt the definition of the 2-chopped convex hull as well as all the results presented in Sections 4 and 5.

6.3. *From Integers to Reals*

While the theory developed in Section 4 is still sound, T(O)RMC does not seem to be able to detect increments when starting from a set that contains non integer numbers. Being able to handle such cases is a direction for future research.

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.

[c]An *increment* between two words w_1 and w_2 is a finite word w_I such that $w_1 = w_{11}w_{12}$ and $w_2 = w_{11}w_Iw_{12}$.

Finite convex polytopes in $\mathbb{Z}^n$			
Vertices	$\lvert A_E \rvert$	$\lvert A_E^i \rvert$	$\lvert A_{Conv(E)} \rvert$
(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 polytopes in $\mathbb{Z}^n$.

In Table 1 we give examples in which the initial set is the set of points in $\mathbb{Z}^n$ within a finite convex polytope described by its vertices. We give the number of states of the RVA that represents each of those sets (first column), of the RVA that represents the largest element in their median sequence (second column), and of the RVA that represents their convex hull (third column). The same information is given for the difference/union of finite convex polytopes in $\mathbb{Z}^n$ in Table 2 (the polytope with vertices $v_1, \dots v_k$ being denoted by $[v_1, \dots, v_k]$). Table 3 gives results for sets described by a the enumeration of their members. Finally, 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 when the convex hull is a closed set. This means that we did not encounter situations where T(O)RMC produced a safe but not precise extrapolation.

All those examples were handled in less than a minute. We observed that 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 $\mathbb{R}^{2n+1}$. This clearly should be fixed to make the approach more practical and, fortunately, [12] shows a promising way of doing this[d]. We also observed that the determinization steps involved in the computation of the successive element of the median sequence are quite costly. We hope to improve the efficiency of those steps by adapting the so called "dominance principle" of [5].

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

[d]Before being able to apply the results in [12] to our problem, we would first have to propose an extrapolation theory for the representation suggested in [12]. One potential difficulty is that increment isomporphism might not be conserved by the "don't cares" introduced in [12].

Non convex sets in $\mathbb{Z}^n$			
Description	$\lvert A_E\rvert$	$\lvert A_E^i\rvert$	$\lvert A_{Conv(E)}\rvert$
[(0,0), (4,4), (8,0)] \ [(4,0), (4,2), (6,0)]	65	97	61
[(0,0), (3,3), (6,3), (6,0)] ∪ [(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)] ∪ [(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)] ∪ [(-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)] \ [(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)] \ [(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 of finite convex polytopes in in $\mathbb{Z}^n$.

Sets of points			
Points of the set	$\lvert A_E\rvert$	$\lvert A_E^i\rvert$	$\lvert A_{Conv(E)}\rvert$
(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.

also grateful to Felix Klaedtke of ETH Zurich for providing insightful comments that have helped improving this work. Finally, we thank Jérôme Leroux of Labri Bordeaux for answering many email questions about his work.

References

[1] 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.

[2] 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.

Infinite sets			
Description	$\lvert A_E \rvert$	$\lvert A_E^i \rvert$	$\lvert A_{Conv(E)} \rvert$
$(y = x \ \cup \ y = -x) \ \cap \ y \geq 0$	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.

[3] B. Boigelot and F. Herbreteau. The power of hybrid acceleration. In *Proc of CAV*, volume 4144 of *LNCS*, pages 438–451. Springer, 2006.
[4] 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.
[5] B. Boigelot, A. Legay, and P. Wolper. Omega-regular model checking. In *Proc of TACAS*, volume 2988 of *LNCS*, pages 561–575. Springer, 2004.
[6] 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.
[7] 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.
[8] 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.
[9] 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.
[10] F. Cantin. Techniques d'extrapolation d'automates: Application au calcul de la fermeture convexe. Master's thesis, University of Liège, Belgium, 2007.
[11] B. Chazelle. An optimal convex hull algorithm in any fixed dimension. *Discrete & Computational Geometry*, 10:377–409, 1993.
[12] 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.
[13] 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.
[14] 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.
[15] M. Krein and D. Milman. On the extreme points of regularly convex sets. *Studia Mathematica*, 9:133–138, 1940.
[16] O. Kupferman and M. Vardi. Weak alternating automata are not that weak. In *Proc. 5th Israeli Symposium on Theory of Computing and Systems*, pages 147–158, Ramat-Gan, Israel, 1997. IEEE Computer Society Press.
[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] A. Legay. *Generic Techniques for the Verification of Infinite-state Systems*. Collection des publications de la Faculté des Sciences Appliquées de l'Université de Liège, Liège, Belgium, 2007. Available at `http://www.montefiore.ulg.ac.be/~legay/papers/index`.

[20] A. Legay. T(O)RMC: A tool for (omega)-regular model checking. In *Proc of CAV08*, LNCS, pages 548–551. Springer, 2008.

[21] J. Leroux. *Algorithmique de la vérification des systèmes à compteurs. Approximation et accélération. Implémentation de l'outil FAST*. PhD thesis, LSV Cachan, France, 2004.

[22] J. Leroux. A polynomial time presburger criterion and synthesis for number decision diagrams. In *Proc of LICS*, pages 147–156. IEEE Computer Society, 2005.

[23] J. Leroux. Convex hull of arithmetic automata. In *Proc of SAS08*, LNCS, pages 47–61. Springer, 2008.

[24] J. Leroux and G. Sutre. Flat counter automata almost everywhere! In *Proc of ATVA*, volume 3707 of *LNCS*, pages 489–503. Springer, 2005.

[25] C. Löding. Efficient minimization of deterministic weak $\omega-$automata. *Information Processing Letters*, 79(3):105–109, 2001.

[26] H. Minkowski. Geometrie der zahlen. *Bulletin of American Mathematical Society*, 21(3):131–132, 1914.

[27] S. Miyano and T. Hayashi. Alternating finite automata on omega-words. *Theoretical Computer Science*, 32:321–330, 1984.

[28] 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.

[29] The T(O)RMC toolset. Available at `http://www.montefiore.ulg.ac.be/~legay/TORMC/index-tormc.html`.

[30] M. Vardi. The Büchi complementation saga. In *Proc of STACS*, volume 4393 of *LNCS*, pages 12–22. Springer, 2007.

[31] 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.