Approximating Integer Solution Counting via Space Quantification for Linear Constraints

Cunjing Ge[1,3], **Feifei Ma**[1,2,3*], **Xutong Ma**[1,3], **Fan Zhang**[1,3], **Pei Huang**[1,3] and **Jian Zhang**[1,3*]

[1]State Key Laboratory of Computer Science, ISCAS

[2]Laboratory of Parallel Software and Computational Science, ISCAS

[3]University of Chinese Academy of Sciences

{gecj, maff, zj}@ios.ac.cn

Abstract

Solution counting or solution space quantification (means volume computation and volume estimation) for linear constraints (LCs) has found interesting applications in various fields. Experimental data shows that integer solution counting is usually more expensive than quantifying volume of solution space while their output values are close. So it is helpful to approximate the number of integer solutions by the volume if the error is acceptable. In this paper, we present and prove a bound of such error for LCs. It is the first bound that can be used to approximate the integer solution counts. Based on this result, an approximate integer solution counting method for LCs is proposed. Experiments show that our approach is over 20x faster than the state-of-the-art integer solution counters. Moreover, such advantage increases with the problem scale.

1 Introduction

Solution Counting, the problem of computing the number of solutions for given constraints, is an important problem with various applications in artificial intelligence including probabilistic inference [Roth, 1996; Chavira and Darwiche, 2008], probabilistic logic programming [Aziz *et al.*, 2015], planning [Domshlak and Hoffmann, 2007], knowledge representation [Lee and Wang, 2018] and privacy/confidentiality verification [Fredrikson and Jha, 2014].

For propositional formulas, the counting problem is known as model counting or #SAT, to which probabilistic inference is easily reducible. #SAT is the canonical #P-complete problem, thus exact counting is intractable for large instances. Recently, the hash-based approximate counters [Chakraborty *et al.*, 2016b; Ge *et al.*, 2018a] attract much attention for improving scalability and providing rigorous guarantees. For Satisfiability Modulo Theories (SMT) formulas, the counting problem is known as #SMT [Chistikov *et al.*, 2015; Chakraborty *et al.*, 2016a; Ge *et al.*, 2018b], which is also an emerging topic in this realm.

As the fundamental constraints that are used in many areas, linear constraints (LCs) have been studied thoroughly. The integer solution counting over LCs is also an important problem and has many applications, such as counting-based search [Zanarini and Pesant, 2007; Pesant, 2016], simple temporal planning [Huang *et al.*, 2018] and probabilistic program analysis [Geldenhuys *et al.*, 2012; Luckow *et al.*, 2014]. This problem is proved to be #P-hard [Valiant, 1979]. In practice, state-of-the-art tools for counting integer solutions on LCs, like `LattE` [Loera *et al.*, 2004], often have difficulty when the number of dimensions is greater than 10 (i.e., when there are more than 10 numeric variables). As a result, they are usually not sufficient for practical applications.

To overcome such problem, we would like to investigate the relation between the integer solution count and the volume of the solution space. Consider the following set of LCs:

$$(a + b > c) \wedge (a + c > b) \wedge (b + c > a) \wedge$$
$$(1 \leq a \leq 32) \wedge (1 \leq b \leq 32) \wedge (1 \leq c \leq 32).$$

The volume of its solution space is 16291 which is close to the number of integer solutions, 16400. From more experiments, we observe that the execution time of space quantification (means volume computation and volume estimation) tools is usually much smaller than the counters, while their output values are close (for details, see Section 5). Naturally we wonder if there exists a relation between the space volume and the solution count, so that space quantification tools can be used to approximate integer solution counts.

In this paper, the first theoretical bound of approximating integer counts via space quantification for LCs is proposed and also proved. Based on these theoretical analysis, we present an approximate counting algorithm for LCs, which can give results with rigorous guarantees. This technique is orthogonal with respect to the current design of model counters and also space quantification tools. As a result, it is easy to apply. We evaluated our approach over random and structural cases generated from various applications. Experiments show that our approach achieves more than 20x speedup compared with state-of-the-art integer solution counters. Moreover, the advantage increases with the number of dimensions. The results also show that our approach could serve as a basis for many applications.

2 Background

The state-of-the-art tools for integer solution counting and space quantification are listed in Table 1. `LattE` [Lo-

*Corresponding Authors

Tool	Algorithm	counting	vol. comput.	vol. est.	guarantee	linear	non-linear	P-time
LattE	Barvinok's Algorithm	✓	×	×	—	✓	×	×
qCORAL	direct Monte-Carlo	×	×	✓	×	✓	✓	×
PolyVest	Multiphase Monte-Carlo	×	×	✓	✓	✓	×	✓
Vinci	Lassere's Method	×	✓	×	—	✓	×	×

Table 1: The state-of-the-art integer solution counter and space quantification tools for numeric constraints

era *et al.*, 2004] is an implementation of Barvinok's algorithm [Barvinok, 1993] dedicated to the counting of integer solutions for LCs. For solving space quantification problems, there are more candidates: `qCORAL` [Borges *et al.*, 2014; 2015], `PolyVest` [Ge and Ma, 2015] and `Vinci` [Büeler *et al.*, 2000]. `qCORAL` is a volume estimation tool, based on direct Monte-Carlo sampling. It can handle arbitrary constraints with real domain, such as non-linear polynomial constraints, trigonometric constraints, etc. `PolyVest` is also a volume estimation tool. It only handles LCs, but provides rigorous guarantees. It is an implementation of Multiphase Monte-Carlo algorithm [Dyer *et al.*, 1991; Lovász and Deák, 2012] with various improvements, whose time complexity is $O^*(n^4)$ (O^* indicates that we suppress factors of $\log n$). Different from previous two tools, `Vinci` is a tool to compute the exact volume of solution space. It consists of several algorithms, but only Lasserre's method is called in this paper as others require inputs of convex hull.

Our experiments on `LattE`, `Vinci` and `PolyVest` show that the integer solution counts are usually very close to the volume for LCs (for details, see Section 5). Similar to a numeric integration procedure, it is trivial to estimate the volume of space with an algorithm based on integer solution counting. The estimation can be arbitrarily precise as we keep differentiating the space. On the contrary, there is no guarantee to approximate the number of integer solutions with space quantification. For example, a very 'thin' rectangle whose both sides are parallel to the coordinates and the short side lies in interval $(0, 1)$. There is no integer point in it, but its volume can be arbitrarily large as the long side stretches. The theoretical result in [Kannan and Vempala, 1997] also shows such phenomenon. So it is necessary to find a bound to quantify the quality of an approximation.

3 Analysis

In this section, we propose and prove our main theoretical results, Theorem 1, 2 and Corollary 1, which provide bounds for approximating counts via volume, on LCs.

3.1 Preliminaries

We first provide the formal definitions of the approximation problem considered in this paper.

Definition 1. *A LC can be written in the form* $a_1x_1 + a_2x_2 + \cdots + a_nx_n$ *op* a_0*, where* x_i *are numeric variables,* a_i *are constant coefficients, and op* $\in \{<, \leq, >, \geq, =\}$*.*

Given a set of LCs F with real variables.

- Let $\#F$ denote the number of integer solutions of F.
- Let $vol(F)$ denote the volume (Lebesgue measure) of solution space of F.

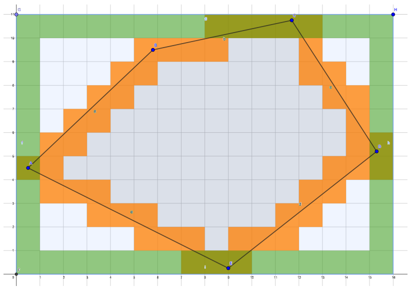

Figure 1: An example about integer cubes.

Problem 1. *Given a set of LCs with real variables, we would like to obtain a bound* **c** *such that* $|vol(F) - \#F| \leq c$*.*

When variables take values from integer domains, the definition of our approximation problem is slightly different. We consider the **relaxed** formula $relaxed(F)$, where $relaxed(F)$ is F whose variables take real values.

Problem 2. *Given a set of LCs with integer variables, we would like to obtain a bound* **c** *such that* $|vol(relaxed(F)) - \#F| \leq c$*.*

A set of LCs also represents a convex polytope P. Similarly, let $\#P$ and $vol(P)$ denote the number of integer points in P and the volume, respectively.

Problem 3. *Given a convex polytope* P*, we would like to obtain a bound* **c** *that* $|vol(P) - \#P| \leq c$*.*

Note that Problem 1, 2 and 3 are all equivalent. Without loss of generality, in this paper, we only use the definitions of problems on LCs with real variables and convex polytopes. In this section, we mainly discuss on convex polytopes. There are several more notations about polytopes.

- Let $B(P)$ denote the boundary (facets) of P.
- Let $m_i(P)$ denote $\lceil min\{x_i | \boldsymbol{x} \in P\} - 1\rceil$. For example, $min\{x_i | \boldsymbol{x} \in P\} = 1$, then $m_i(P) = 0$.
- Let $M_i(P)$ denote $\lfloor max\{x_i | \boldsymbol{x} \in P\} + 1\rfloor$ similarly.
- Let P_h denote the intersection of P with a hyperplane h. For example, $P_{x_1=0}$ represents the cross-section of P with $x_1 = 0$, i.e., $P_{x_1=0} = \{\boldsymbol{Ax} \leq \boldsymbol{b}, x_1 = 0\}$.
- Let $proj_h(P)$ denote the projection from P onto a hyperplane h.

Definition 2. *An* ***integer-cube*** *is a unique unit-cube all of whose corners are integer points.*

- Given an integer point $\boldsymbol{\alpha}$, we use $c(\boldsymbol{\alpha})$ to denote the integer-cube, where $c(\boldsymbol{\alpha}) = \{x_i \in [\alpha_i, \alpha_i + 1], i = 1, \ldots, n\}$.

- We use $C(P)$ to denote the **set** of integer-cubes which intersects with P. For example, $C(P)$ is the set of orange squares and grey squares in Figure 1. Similarly, we use $C(B(P))$ and $C(proj_h(P))$ to denote sets of integer-cubes that intersect with $B(P)$ and $proj_h(P)$ respectively.

3.2 Main Results

The difference between $vol(P)$ and $\#P$ is intuitively caused by points that are close to the boundary of P. Moreover, these "controversial" points can be covered by integer-cubes that intersect with $B(P)$, e.g., orange squares in Figure 1. So we draw the following theorem by this intuition.

Theorem 1. $|vol(P) - \#P| \leq \big|C(B(P))\big|$.[1]

Proof. Given an integer-cube $c \in C(P) \setminus C(B(P))$. Since $c \in C(P)$, c has intersection with P. On the other hand, since $c \notin C(B(P))$, c does not has intersection with the boundary $B(P)$. So we know $c \subset P$, i.e., $C(P) \setminus C(B(P))$ is a set of integer-cubes inside P. Note that integer-cubes only share points on their boundaries and the volume of an integer-cube is 1. We have $\big|C(P) \setminus C(B(P))\big| = \sum_{c \in C(P)\setminus C(B(P))} vol(c) \leq vol(P)$. We can map cubes in $C(P) \setminus C(B(P))$ to different integer points in P, e.g., pick a specific corner point of a cube. Therefore, the number of cubes $\big|C(P) \setminus C(B(P))\big|$ is less or equal to $\#P$. Note that $\big|C(P) \setminus C(B(P))\big| = |C(P)| - \big|C(B(P))\big|$. Then we obtain

$$\begin{cases} |C(P)| - \big|C(B(P))\big| \leq vol(P) \leq |C(P)|, \\ |C(P)| - \big|C(B(P))\big| \leq \#P \leq |C(P)|. \end{cases}$$

As a result, $vol(P) - \big|C(B(P))\big| \leq |C(P)| - \big|C(B(P))\big| \leq \#P \leq |C(P)| \leq vol(P) + \big|C(B(P))\big|$. ☐

In Figure 1, the size of the orange area is equal to $\big|C(B(P))\big|$, which consists of integer-cubes in $C(B(P))$. Theorem 1 is the basis of theoretical guarantees, but lacks the bound of $\big|C(B(P))\big|$. So it is our goal in the following analysis.

Definition 3. *We call $c(\boldsymbol{\alpha})_{\{x_n=\alpha_n\}}$, which is a cross-section, the **bottom** of the integer-cube $c(\boldsymbol{\alpha})$, denoted as $bottom(c(\boldsymbol{\alpha}))$.*

Definition 4. *We call a polytope P **cuts** an integer-cube c if $B(P) \cap c \neq \emptyset$ and $B(P) \cap bottom(c) = \emptyset$, i.e., $B(P)$ intersects with c but does not intersects with $bottom(c)$.*

Lemma 1. *Consider a set of integer-cubes $S_n(\boldsymbol{\alpha}) = \{c(\boldsymbol{\alpha_\gamma}) | \boldsymbol{\alpha_\gamma} = (\alpha_1, \ldots, \alpha_{n-1}, \gamma), \gamma = \lfloor m_n \rfloor, \ldots, \lceil M_n \rceil\}$. Then P can cut **at most two** integer-cubes in $S_n(\boldsymbol{\alpha})$.*

Proof. Assume there are three integer-cubes $c_1, c_2, c_3 \in S_n(\boldsymbol{\alpha})$ such that $B(P) \cap c_i \neq \emptyset$ and $B(P) \cap bottom(c_i) = \emptyset$.

Let $\boldsymbol{\alpha_1}$, $\boldsymbol{\alpha_2}$ and $\boldsymbol{\alpha_3}$ denote the points such that $c(\boldsymbol{\alpha_i}) = c_i$, $i = 1, 2, 3$. Note $\boldsymbol{\alpha_i}$s are same except the nth elements. Let v_i represent the value of nth element in $\boldsymbol{\alpha_i}$, i.e., $\boldsymbol{\alpha_i} = (\alpha_1, ..., \alpha_{n-1}, v_i)$. Without loss of generality, let $v_1 < v_2 < v_3$. Then c_1 is above c_2 and c_2 is above c_3.

Since $B(P) \cap c_1 \neq \emptyset$ and $B(P) \cap c_2 \neq \emptyset$, we can pick two points $\boldsymbol{a} \in B(P) \cap c_1$ and $\boldsymbol{b} \in B(P) \cap c_2$. The segment $\boldsymbol{ab}$ is inside P due to the convexity of P. In addition, $\boldsymbol{ab} \cap bottom(c_1) \neq \emptyset$ as c_1 is above c_2. So $bottom(c_1) \cap P \neq \emptyset$. Furthermore, we have $bottom(c_1) \subset P$, otherwise, there exists a point in $bottom(c_1)$ that also lies in $B(P)$, which contradicts with $B(P) \cap bottom(c_1) = \emptyset$. Similarly, we can prove $bottom(c_2) \subset P$, because $B(P) \cap c_2 \neq \emptyset$, $B(P) \cap c_3 \neq \emptyset$, and c_2 is above c_3.

From the assumption $c_2 \cap B(P) \neq \emptyset$, one can pick a point $\boldsymbol{p} \in c_2 \cap B(P)$, where $\boldsymbol{p} = (p_1, \ldots, p_n)$. Consider two points $\boldsymbol{p'} = (p_1, \ldots, p_{n-1}, v_1)$ and $\boldsymbol{p''} = (p_1, \ldots, p_{n-1}, v_2)$, obviously, $\boldsymbol{p'} \in bottom(c_1)$ and $\boldsymbol{p''} \in bottom(c_2)$. Moreover, $bottom(c_1) \subset P$ and $B(P) \cap bottom(c_1) = \emptyset$, so we have $\boldsymbol{p'} \in P \setminus B(P)$. Similarly, we have $\boldsymbol{p''} \in P \setminus B(P)$. From the convexity, the segment $\boldsymbol{p'p''} \subset P \setminus B(P)$. We know that $\boldsymbol{p}$ is on segment $\boldsymbol{p'p''}$, then $\boldsymbol{p}$ is not in the boundary of P, which is a contradiction. ☐

Intuitively, we could obtain the bound of $\big|C(B(P))\big|$ by mapping integer-cubes in $C(B(P))$ to integer-cubes which intersect with a bounding cuboid of P. For example, orange squares can be mapped to green squares in Figure 1. In detail, we prove the following theorem.

Theorem 2.

$$\big|C(B(P))\big| \leq 2 \sum_{i=1}^{n} \big|C(proj_{\{x_i=0\}}(P))\big|. \tag{1}$$

Proof. (Mathematical Induction)

Basis: Show that the Equation (1) holds for 1-dimensional problems. A 1-dimensional polytope is an interval. Its boundaries are the two end-points of this interval. Then $\big|C(B(P))\big| \leq 2$ and $\big|C(proj_{\{x=0\}}(P))\big| = 1$.

Inductive Step: Show that if Equation (1) holds for $1, 2, \ldots, n-1$ dimensions, then it also holds for n dimensions.

Given a polytope P, we have the length of x_i, i.e., $[m_i, M_i]$. Then we subdivide P by constraints $X_n(w) = \{w \leq x_n \leq w + 1\}$, where $w = \lfloor m_n \rfloor, \ldots, \lceil M_n \rceil - 1$ are integers. It is easy to see that

$$\big|C(B(P))\big| = \sum_{w=\lfloor m_n \rfloor}^{\lceil M_n \rceil - 1} \Big|C\Big(B(P) \cap X_n(w)\Big)\Big|, \tag{2}$$

$$\big|C(\underset{\{x_i=0\}}{proj}(P))\big| = \sum_{w=\lfloor m_n \rfloor}^{\lceil M_n \rceil - 1} \Big|C\Big(\underset{\{x_i=0\}}{proj}(P) \cap X_n(w)\Big)\Big|, \tag{3}$$

where $i = 1, \ldots, n-1$.

Note that $P_{\{x_n=w\}}$ is an $(n-1)$-dimensional polytope. According to the induction hypothesis, there is

$$\big|C(B(P_{\{x_n=w\}}))\big| \leq 2 \sum_{i=1}^{n-1} \Big|C\big(\underset{\{x_i=0\}}{proj}(P_{\{x_n=w\}})\big)\Big|. \tag{4}$$

[1]Note that $\big|C(B(P))\big|$ represents the cardinality of the set $C(B(P))$.

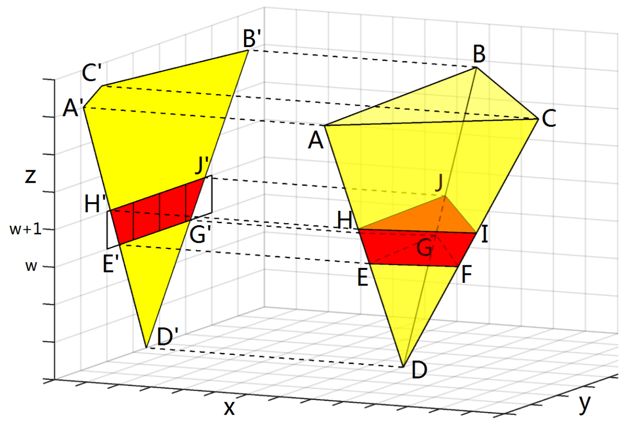

Figure 2: Consider a 3D example of polyhedron $P = ABCD$. We have $P \cap Z(w)$ is the red polyhedron $EFGHIJ$ and $P_{\{z=w\}} = \triangle EFG$, where $Z(w) = \{w \leq z \leq w+1\}$. Then consider the projections onto plane $x = 0$. We have $proj_{\{x=0\}}(P \cap Z(w)) = E'G'J'H'$ and $proj_{\{x=0\}}(P_{\{z=w\}}) = E'G'$. We observe that the trapezoid $E'G'J'H'$ can be covered by 4 integer-cubes (squares) and $E'G'$ can be covered by 4 integer-cubes (line segments).

Since $P_{\{x_n=w\}} \subseteq P \cap X_n(w)$, it is easy to see that

$$\left|C\big(\underset{\{x_i=0\}}{proj}(P_{\{x_n=w\}})\big)\right| \leq \left|C\big(\underset{\{x_i=0\}}{proj}(P \cap X_n(w))\big)\right|.$$

We know that $proj_h(A \cap B) \subseteq proj_h(A) \cap proj_h(B)$ and $proj_{\{x_i=0\}}(P) \cap proj_{\{x_i=0\}}(X_n(w)) = proj_{\{x_i=0\}}(P) \cap X_n(w)$, so we have

$$\left|C\big(\underset{\{x_i=0\}}{proj}(P \cap X_n(w))\big)\right| \leq C\left|\big(\underset{\{x_i=0\}}{proj}(P) \cap X_n(w)\big)\right|.$$

Then Equation (4) becomes

$$\left|C(B(P_{\{x_n=w\}}))\right| \leq 2\sum_{i=1}^{n-1}\left|C\big(\underset{\{x_i=0\}}{proj}(P) \cap X_n(w)\big)\right|. \quad (5)$$

In order to have a better understanding of this proof, we also present a simple 3-dimensional example in Figure 2.

Let $F_w = \left|C\Big(B(P) \cap X_n(w)\Big)\right| - \left|C(B(P_{\{x_n=w\}}))\right|$. Then F_w is the number of integer-cubes that $B(P)$ has intersections with, but no intersection with their bottoms, in $C\big(B(P) \cap X_n(w)\big)$. For example, in Figure 2, $B(P_{\{z=w\}})$ are the 3 sides of $\triangle EFG$ and $B(P) \cap Z(w)$ are the 3 quadrangles $EFIH$, $FGJI$ and $EGJH$. It is easy to see that an integer-square $s \in C\big(B(P_{\{z=w\}})\big)$ corresponds to a unique integer-cube $c \in C\big(B(P) \cap Z(w)\big)$ that $s = (c \cap (z = w))$. Then $s = bottom(c)$ and $(s \cap B(P)) \neq \emptyset$. It implies that $bottom(c)$ intersects with $B(P)$. On the other hand, we can also find that each integer-cube in $C\big(B(P) \cap Z(w)\big)$, whose bottom intersects with $B(P)$, corresponds to a unique integer-square in $C\big(B(P_{\{z=w\}})\big)$. So $\left|C\big(B(P_{\{z=w\}})\big)\right|$ is equal to the number of integer-cubes in $\left|C\big(B(P) \cap Z(w)\big)\right|$ whose bottoms have intersection with $B(P)$.

Combining Equation (2), (3) and (5), we have

$$\begin{aligned}
\left|C(B(P))\right| &= \sum_{w=\lfloor m_n \rfloor}^{\lceil M_n \rceil - 1}\Big(\left|C(B(P_{\{x_n=w\}}))\right| + F_w\Big) \\
&\leq 2\sum_{i=1}^{n-1}\sum_{w=\lfloor m_n \rfloor}^{\lceil M_n \rceil - 1}\left|C\big(\underset{\{x_i=0\}}{proj}(P) \cap X_n(w)\big)\right| + \sum_{w=\lfloor m_n \rfloor}^{\lceil M_n \rceil - 1} F_w, \\
&\leq 2\sum_{i=1}^{n-1}\left|C(proj_{\{x_i=0\}}(P))\right| + \sum_{w=\lfloor m_n \rfloor}^{\lceil M_n \rceil - 1} F_w.
\end{aligned}$$

From Lemma 1, we can map each integer-cube, which $B(P)$ has intersections with, but no intersection with their bottoms, to a cube in $C(proj_{\{x_n=0\}}(P))$ by at most two times. So

$$\sum_{w=\lfloor m_n \rfloor}^{\lceil M_n \rceil - 1} F_w \leq 2\left|C(proj_{\{x_n=0\}}(P))\right|.$$

□

Since computing the projection is not an easier problem, we would like to consider a simpler and practical version of Theorem 2, which is the following corollary.

Corollary 1.

$$\left|C(B(P))\right| \leq 2\sum_{i=1}^{n}\prod_{i \neq j}\big(M_j(P) - m_j(P)\big). \quad (6)$$

We omit the proof of Corollary 1 which is plain, due to the limited space. This corollary provides a bound for $\left|C(B(P))\right|$ by volumes of surfaces of a bounding cuboid for P. To obtain the values of $M_i(P)$ and $m_i(P)$, it requires at most $2n$ times of linear programming. In practice, the overhead is negligible compared with the cost of volume computation or estimation.

4 Algorithm

Based on Theorem 1 and Corollary 1, we present an approximation algorithm for integer solution counting with rigorous guarantees on LCs. The pseudo-code of our approach `Vol2Lat` is presented in Algorithm 1. The input is a convex polytope P (or a set of LCs) and the output is a triple that consists of the value, the lower bound and the upper bound of approximation.

We first apply Gaussian Elimination to reduce equalities, as they usually cause degeneration and lead to zero volume. Then we compute M_i and m_i for P_k by linear programming and err_k via the formula in Corollary 1. Since there are volume computation tools and volume estimation tools for convex polytopes, whose outputs are different, we present two procedures for them. For the volume v obtained by computation, the lower and upper bound are $v - err$ and $v + err$ respectively. Different from volume computation, the estimation routine, e.g., `PolyVest`, returns results in (ϵ, δ)-bound. It means that the estimation $\tilde{v}$ of $vol(P)$ lies in the interval $[(1+\epsilon)^{-1}vol(P), (1+\epsilon)vol(P)]$ with at least probability $(1-\delta)$. Such interval is also called the confidence interval. Let $[I_l, I_u]$ denote such $(1-\delta)$ confidence interval. Then the bound is thus equal to $[I_l - err_k, I_u + err_k]$, which is also an $(1-\delta)$ confidence interval.

Algorithm 1 Approximate Counting via Volume

Function *Vol2Lat*(P)
  Apply Gaussian Elimination;
  Compute $M_i(P)$ and $m_i(P)$, $i = 1, \ldots, n$;
  $err \leftarrow 2 \sum_{i=1}^{n} \prod_{i \neq j} \left(\lceil M_j(P) \rceil - \lfloor m_j(P) \rfloor \right)$;
  if *enable volume computation* **then**
    Call the volume computation routine to obtain the exact volume v of P;
    return $(v, v - err, v + err)$;
  else
    Call the volume estimation routine to obtain the estimation $\tilde{v}$ with the confidence interval $[I_l, I_u]$;
    return $(\tilde{v}, I_l - err, I_u + err)$;

5 Evaluation

In order to evaluate our approach, we implemented a tool called `Vol2Lat`[2]. in C++ based on Algorithm 1 and a #SMT(LA) solver, which can handle LCs combined with propositional logic and also disequalities ($\neq$). We used a **timeout** of **half an hour**, and chose $\epsilon = 0.2$ and $\delta = 0.1$ for `PolyVest` (volume estimation). Experiments were conducted on a laptop workstation with 2.40GHz Intel Core i7-4700MQ CPU and 8GB memory. The suite of benchmarks consists of three families:

- **Random Polytopes:** For each LC, we first randomly pick a number l. Then we select l variables and generate coefficients (from -1000 to 1000) in uniform distribution. We filter out unsat cases and obtain 96 polytopes ranging from 2 to 15 dimensions at last.
- **Multiple Knapsack:** We generated 100 instances from the classic multiple knapsack problem [Abío and Stuckey, 2014]. The coefficients belong to $[0, 10]$, the domain size is 64 and $m = n, n \in \{3, 4, 5\}$.
- **Simple Temporal Planning:** We randomly generated 150 instances based on the structure of Simple Temporal Networks (STNs) [Huang *et al.*, 2018].
- **Instances from program analysis:** We analyzed 7 programs ('cubature', 'gjk', 'http-parser', 'muFFT', 'SimpleXML', 'tcas' and 'timeout') ranging from $0.4k$ to $7.7k$ lines of source code via a symbolic execution bug-finding tool called `CAnalyze` [Xu *et al.*, 2014][3] and generated 3803 instances in total.

Experimental Results

Figure 3 presents the average running time of three tools for each dimension. Note that the y-axis is in logarithmic coordinate. Table 2 presents more details, but only part of results, due to the limited space. $\#dims$ is the number of dimensions. $L(F)$ and $U(F)$ are the lower and upper bound computed by our approach. 'TIMEOUT' and 'MEMOUT' represent running out of time and memory (8GB) respectively.

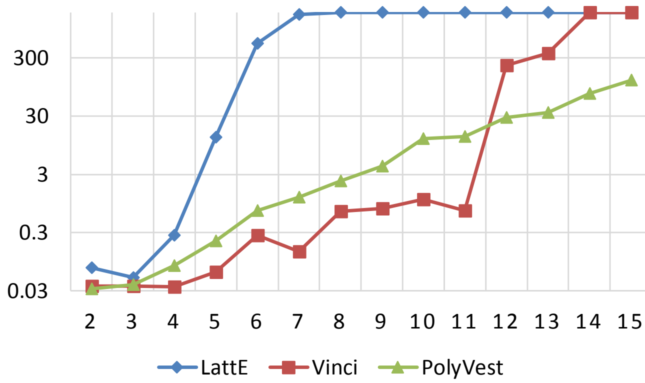

Figure 3: Average running times of `LattE`, `Vinci` and `PolyVest` on random polytopes, for each dimension.

The results show that `PolyVest` and `Vinci` significantly outperform `LattE`. The running time of `LattE` exponentially increases as dimension increases. `Vinci` is very fast but suffers from tremendous memory consumption. However, the running time of `PolyVest` increases mildly. Due to the properties of the algorithms of `LattE` and `Vinci`, their running times are heavily affected by the structure of LCs. On some structural cases, `LattE` and `Vinci` can handle LCs with more dimensions and usually outperform `PolyVest`. However, since `PolyVest` employs a polynomial-time algorithm, it will eventually gain the upper hand. Besides the time-consumption, we can also observe that the values of $vol(F)$ and $\tilde{vol}(F)$ are very close to $\#F$ from Table 2. It is the key observation that motivates us to approximate integer counts via space volume. In addition, the bounds are usually close to $\#F$. Recall that in Theorem 1, we compute bounds by subtracting or adding $|C(B(P))|$ from $\#P$. So, the lower bounds are worse and sometimes may be equal to zero.

To evaluate the quality of approximation bounds in applications, we conducted more experiments over constraints generated from multiple knapsack problems, STNs and program analysis. The average results for each family of benchmarks are listed in Table 3. Note that $\bar{e}$, $\bar{e}_l$ and $\bar{e}_u$ are the average values of $e = \frac{|\#F - vol(F)|}{\#F}$, $e_l = \frac{L(F)}{\#F}$ and $e_u = \frac{U(F)}{\#F}$ respectively. In general, the relative errors of approximations via volume are smaller than 10% and the bounds are often close to the exact counts #F, which are useful in many applications, such as counting-based search, knowledge representation, program analysis, etc. Since in these cases, we usually only need to find the polytope which has larger solution count instead of obtaining the exact count. For example, comparing the solution counts of branches to guide decision procedure during the search.

6 Related Works

Kannan and Vempala [Kannan and Vempala, 1997] proposed an algorithm for sampling integer points in a convex polytope.

[2] Our tool and benchmarks are available at: https://github.com/bearben/VOL2LAT

[3] In order to analyze practical programs, bug-finding tools employ simplifications. For example, they unwind a loop by once (or several times), do not instantiate every element in an array, etc.

#dims	Exact Counting (`LattE`)		Approx via Volume Computation (`Vinci`)				Approx via Volume Estimation (`PolyVest`)			
	$\#F$	t (s)	$vol(F)$	$L(F)$	$U(F)$	t (s)	$\widetilde{vol}(F)$	$\tilde{L}(F)$	$\tilde{U}(F)$	t (s)
2	1.59e+9	0.030	1.59e+9	1.59e+9	1.59e+9	0.032	1.58e+9	1.32e+9	1.90e+9	0.029
3	2.07e+13	0.040	2.07e+13	2.07e+13	2.07e+13	0.027	2.11e+13	1.76e+13	2.54e+13	0.025
4	5.29e+17	0.134	5.29e+17	5.29e+17	5.30e+17	0.031	5.27e+17	4.39e+17	6.33e+17	0.052
5	4.63e+19	2.059	4.63e+19	4.50e+19	4.76e+19	0.025	4.60e+19	3.70e+19	5.65e+19	0.146
6	1.05e+26	704.7	1.05e+26	1.03e+26	1.08e+26	0.027	1.06e+26	8.59e+25	1.29e+26	0.444
7	——	TIMEOUT	4.29e+31	4.24e+31	4.35e+31	0.028	4.26e+31	3.49e+31	5.17e+31	0.894
8	——	TIMEOUT	1.08e+35	9.47e+34	1.21e+35	0.061	1.09e+35	7.75e+34	1.43e+35	1.881
9	——	TIMEOUT	1.28e+41	1.27e+41	1.29e+41	0.050	1.27e+41	1.05e+41	1.53e+41	4.451
10	——	TIMEOUT	2.66e+44	1.70e+44	3.62e+44	0.479	2.69e+44	1.29e+44	4.19e+44	6.721
11	——	TIMEOUT	1.29e+48	0	3.18e+48	0.354	1.28e+48	0	3.42e+48	11.3
12	——	TIMEOUT	4.39e+53	2.80e+53	5.97e+53	4.944	4.38e+53	2.07e+53	6.83e+53	21.4
13	——	TIMEOUT	——	——	——	MEMOUT	1.28e+58	1.96e+57	2.41e+58	35.4
14	——	TIMEOUT	——	——	——	MEMOUT	4.50e+63	2.02e+62	8.96e+63	56.8
15	——	TIMEOUT	——	——	——	MEMOUT	2.73e+64	0	4.73e+66	70.1

Table 2: Running times and approximation results over random polytopes with different dimensions

Name	#dims		Exact Counting		Approximating via Volume Computation				
	avg	max	#solved	$\bar{t}$ (s)	#solved	$\bar{t}$ (s)	$\bar{e}$ ($\geq 0\%$)	$\bar{e}_l$ ($\leq 100\%$)	$\bar{e}_u$ ($\geq 100\%$)
Knapsack	5	7	100 (100)	80.8	100 (100)	0.04	7.7%	35.9%	150.6%
STN	8.71	14	150 (150)	16.4	148 (150)	0.73	3.8%	77.9%	114.9%
cubature	7.50	14	1094 (1099)	52.3	1099 (1099)	1.28	5.9%	14.1%	336.0%
gjk	6.83	10	58 (58)	2.46	58 (58)	0.09	$< 0.1\%$	0%	806.9%
http-parser	5.36	16	1660 (1710)	64.1	1710 (1710)	2.48	1.4%	90.5%	117.7%
muFFT	5.04	10	593 (593)	0.67	593 (593)	0.05	0.5%	45.6%	500.6%
SimpleXML	4.87	8	113 (113)	2.54	113 (113)	0.10	3.4%	87.2%	160.4%
tcas	7.15	13	107 (107)	7.83	107 (107)	0.35	$< 0.1\%$	$> 99.9\%$	$< 100.1\%$
timeout	7.97	13	122 (123)	171.3	123 (123)	3.08	4.0%	92.9%	118.7%

Table 3: Statistic results of time-consumption and errors of approximation and bounds on structural instances.

The proof of this algorithm shows that if a convex polytope P contains within it a ball of radius $n\sqrt{\log m}$, then there is a constant c such that $c \cdot \#P < vol(P) < \frac{1}{c}\#P$. Intuitively, the bound of c can be possibly approximated by inverting this sampling algorithm. However, there is no expression nor algorithm proposed to compute the constant c.

The Ehrhart Polynomial, to our best knowledge, is the result that is closest to our bounds. It says that $\#P$ can be expressed as a linear combination of the volume of the faces of P, i.e., $\#P = \sum_F (vol(F) * e(cone(P, F))$, where F is a face of P, $cone(P, F)$ is the supporting cone of P at F. Barvinok [Barvinok, 1994] proved that for integer polytopes, the coefficient $e(cone(P, F))$ can be computed using $O(m^k)$ calls for the volume computation oracle, where m is the number of faces and F is k-dimensional. For rational polytopes, coefficients are considered to be more difficult to compute.

Dyer [Dyer *et al.*, 1991] introduced a FPRAS algorithm for estimating the volume of convex bodies which is called Multiphase Monte-Carlo by its follow-up works. `PolyVest` is an implementation of such algorithm, which is restricted to polytopes for efficiency. But the goal of our work is different since we consider the integer solution counts instead of volume. Recall the discussion in Section 2, discretization will not work on integer counting as the length of grids cannot be arbitrarily small, although it is a natural way to handle continuous problems like volume.

7 Conclusion

Linear constraints appear frequently in many areas. Yet the associated counting problem has not received enough attention. We introduced an approximate counting algorithm via space quantification for LCs. It provides approximation with theoretical guarantees which are useful for users to make decisions of keeping or discarding approximations. Evaluation shows the scalability and high-efficiency of our approach. We also observe that the theoretical bounds are sometimes too loose compared with the practical errors. Therefore, improving bounds, especially the lower bound, is an interesting and also challenging direction of future research.

Acknowledgments

This work has been supported by Key Research Program of Frontier Sciences, Chinese Academy of Sciences (CAS) under Grant QYZDJ-SSW-JSC036. Feifei Ma is also supported by the Youth Innovation Promotion Association, CAS. The authors would like to thank the anonymous reviewers for their comments and suggestions.

References

[Abío and Stuckey, 2014] Ignasi Abío and Peter J. Stuckey. Encoding linear constraints into SAT. In *Proc. of CP*, pages 75–91, 2014.

[Aziz *et al.*, 2015] Rehan Abdul Aziz, Geoffrey Chu, Christian J. Muise, and Peter J. Stuckey. Stable model counting and its application in probabilistic logic programming. In *Proc. of AAAI*, pages 3468–3474, 2015.

[Barvinok, 1993] Alexander I. Barvinok. Computing the volume, counting integral points, and exponential sums. *Discrete & Computational Geometry*, 10:123–141, 1993.

[Barvinok, 1994] Alexander I. Barvinok. Computing the ehrhart polynomial of a convex lattice polytope. *Discrete & Computational Geometry*, 12:35–48, 1994.

[Borges *et al.*, 2014] Mateus Borges, Antonio Filieri, Marcelo d'Amorim, Corina S. Pasareanu, and Willem Visser. Compositional solution space quantification for probabilistic software analysis. In *Proc. of PLDI*, pages 123–132, 2014.

[Borges *et al.*, 2015] Mateus Borges, Antonio Filieri, Marcelo d'Amorim, and Corina S. Pasareanu. Iterative distribution-aware sampling for probabilistic symbolic execution. In *Proc. of ESEC/FSE*, pages 866–877, 2015.

[Büeler *et al.*, 2000] Benno Büeler, Andreas Enge, and Komei Fukuda. *Exact Volume Computation for Polytopes: A Practical Study*, pages 131–154. Birkhäuser Basel, 2000.

[Chakraborty *et al.*, 2016a] Supratik Chakraborty, Kuldeep S. Meel, Rakesh Mistry, and Moshe Y. Vardi. Approximate probabilistic inference via word-level counting. In *Proc. of AAAI*, pages 3218–3224, 2016.

[Chakraborty *et al.*, 2016b] Supratik Chakraborty, Kuldeep S. Meel, and Moshe Y. Vardi. Algorithmic improvements in approximate counting for probabilistic inference: From linear to logarithmic SAT calls. In *Proc. of IJCAI*, pages 3569–3576, 2016.

[Chavira and Darwiche, 2008] Mark Chavira and Adnan Darwiche. On probabilistic inference by weighted model counting. *Artif. Intell.*, 172(6-7):772–799, 2008.

[Chistikov *et al.*, 2015] Dmitry Chistikov, Rayna Dimitrova, and Rupak Majumdar. Approximate counting in SMT and value estimation for probabilistic programs. In *Proc. of TACAS*, pages 320–334, 2015.

[Domshlak and Hoffmann, 2007] Carmel Domshlak and Jörg Hoffmann. Probabilistic planning via heuristic forward search and weighted model counting. *J. Artif. Intell. Res. (JAIR)*, 30:565–620, 2007.

[Dyer *et al.*, 1991] Martin E. Dyer, Alan M. Frieze, and Ravi Kannan. A random polynomial time algorithm for approximating the volume of convex bodies. *J. ACM*, 38(1):1–17, 1991.

[Fredrikson and Jha, 2014] Matthew Fredrikson and Somesh Jha. Satisfiability modulo counting: a new approach for analyzing privacy properties. In *Proc. of CSL-LICS*, pages 42:1–42:10, 2014.

[Ge and Ma, 2015] Cunjing Ge and Ffeifei Ma. A fast and practical method to estimate volumes of convex polytopes. In *Proc. of FAW*, pages 52–65, 2015.

[Ge *et al.*, 2018a] Ccunjing Ge, Feifei Ma, Tian Liu, Jian Zhang, and Xutong Ma. A new probabilistic algorithm for approximate model counting. In *Proc. of IJCAR*, pages 312–328, 2018.

[Ge *et al.*, 2018b] Cunjing Ge, Feifei Ma, Peng Zhang, and Jian Zhang. Computing and estimating the volume of the solution space of SMT(LA) constraints. *Theor. Comput. Sci.*, 743:110–129, 2018.

[Geldenhuys *et al.*, 2012] Jaco Geldenhuys, Matthew B. Dwyer, and Willem Visser. Probabilistic symbolic execution. In *Proc. of ISSTA*, pages 166–176, 2012.

[Huang *et al.*, 2018] Amy Huang, Liam Lloyd, Mohamed Omar, and James C. Boerkoel. New perspectives on flexibility in simple temporal planning. In *Proc. of ICAPS*, pages 123–131, 2018.

[Kannan and Vempala, 1997] Ravi Kannan and Santosh Vempala. Sampling lattice points. In *Proc. of STOC*, pages 696–700, 1997.

[Lee and Wang, 2018] Joohyung Lee and Yi Wang. Weight learning in a probabilistic extension of answer set programs. In *KR 2018*, page to appear, 2018.

[Loera *et al.*, 2004] Jesús A. De Loera, Raymond Hemmecke, Jeremiah Tauzer, and Ruriko Yoshida. Effective lattice point counting in rational convex polytopes. *J. Symb. Comput.*, 38(4):1273–1302, 2004.

[Lovász and Deák, 2012] László Lovász and István Deák. Computational results of an $O^*(n^4)$ volume algorithm. *European Journal of Operational Research*, 216(1):152–161, 2012.

[Luckow *et al.*, 2014] Kasper Søe Luckow, Corina S. Pasareanu, Matthew B. Dwyer, Antonio Filieri, and Willem Visser. Exact and approximate probabilistic symbolic execution for nondeterministic programs. In *Proc. of ASE*, pages 575–586, 2014.

[Pesant, 2016] Gilles Pesant. Counting-based search for constraint optimization problems. In *Proc. of AAAI*, pages 3441–3448, 2016.

[Roth, 1996] Dan Roth. On the hardness of approximate reasoning. *Artif. Intell.*, 82(1-2):273–302, 1996.

[Valiant, 1979] Leslie G. Valiant. The complexity of enumeration and reliability problems. *SIAM J. Comput.*, 8(3):410–421, 1979.

[Xu *et al.*, 2014] Zhenbo Xu, Jian Zhang, Zhongxing Xu, and Jiteng Wang. Canalyze: a static bug-finding tool for C programs. In *Proc. of ISSTA*, pages 425–428, 2014.

[Zanarini and Pesant, 2007] Alessandro Zanarini and Gilles Pesant. Solution counting algorithms for constraint-centered search heuristics. In *Proc. of CP*, pages 743–757, 2007.