Linear Reachability Problems and Minimal Solutions to Linear Diophantine Equation Systems

Gaoyan Xie, Cheng Li and Zhe Dang [1]

School of Electrical Engineering and Computer Science
Washington State University
Pullman, WA 99164, USA

Abstract

The linear reachability problem for finite state transition systems is to decide whether there is an execution path in a given finite state transition system such that the counts of labels on the path satisfy a given linear constraint. Using some known results on minimal solutions (in nonnegative integers) for linear Diophantine equation systems, we present new time complexity bounds for the problem. In contrast to the previously known results, the bounds obtained in this paper are polynomial in the size of the transition system in consideration, when the linear constraint is fixed. The bounds are also used to establish a worst-case time complexity result for the linear reachability problem for timed automata.

Key words: Model-checking, timed automata, reachability, linear Diophantine equation systems, minimal solutions

1 Introduction

Model-checking [7,21] is a technique that automatically verifies a finite state transition system against a temporal property usually specified in, e.g., Computation Tree Logic (CTL) [7] or Linear Temporal Logic (LTL) [19], by exhaustively exploring the finite state space of the system. The usefulness of model-checking has been demonstrated by several model-checkers (e.g., SMV [17], SPIN [15], BMC [3]), which have been successfully used to verify/test industrial-level hardware/software systems with significant sizes.

Although both CTL and LTL are expressive, many temporal properties are out of their scope. For instance, event counting is a fundamental concept to specify some

[1] Corresponding author (zdang@eecs.wsu.edu).

important fairness properties. As a motivating example, we consider the design (depicted as a finite state transition system $\mathbb{A}$ in Figure 1) of a process scheduler. The scheduler schedules two kinds of processes: P_r and P_w according to some scheduling strategy. A transition with label P_r (resp. P_w) is taken when the scheduler chooses a P_r (resp. P_w) process to run. It is required that the design shall satisfy some fairness properties; e.g., starting from state s_0, whenever s_0 is reached, the number of P_r processes scheduled is greater than or equal to the number of P_w processes scheduled and less than or equal to twice the number of P_w processes scheduled. To ensure that the design meets the requirement, we need to check whether for any path p that starts from and ends with s_0, the linear constraint, $\#_{P_w}(p) \leq \#_{P_r}(p) \leq 2\#_{P_w}(p)$, is satisfied, where $\#_{P_w}(p)$ (resp. $\#_{P_r}(p)$) stands for the count of labels P_w (resp. P_r) on path p. Notice that this property is nonregular [6] and, since the counts could go unbounded, the property is not expressible in CTL or LTL.

In general, by considering its negation, the property can be formulated as the *linear reachability problem for finite state transition systems* as follows.

- **Given:** A finite state transition system $\mathbb{A}$ with labels $a_1, \ldots, a_k$, two designated states s_{init} and s_{final}, and a linear constraint $U(x_1, \ldots, x_k)$.
- **Question:** Is there a path p of $\mathbb{A}$ from s_{init} to s_{final} such that p satisfies U (i.e., $U(\#_{a_1}(p), \ldots, \#_{a_k}(p))$ holds)?

The reachability problem is decidable. To see this, one can treat $\mathbb{A}$ as a finite automaton with initial state s_{init} and final state s_{final}. Then, a naive decision procedure can be constructed in the following three steps: *(i).* Compute a regular expression for the regular language (over alphabet $\{a_1, \ldots, a_k\}$) accepted by $\mathbb{A}$, *(ii).* Calculate the semilinear set of the regular expression defined by a Presburger formula R [18], and *(iii).* Check the satisfiability of the Presburger formula $R \wedge U$. Unfortunately, the time complexity of this procedure is at least $O(2^{|S|})$, where $|S|$ is the number of states in $\mathbb{A}$, even when k is fixed. This is because the size of the regular expression, in worst cases, is exponential in $|S|$ [16].

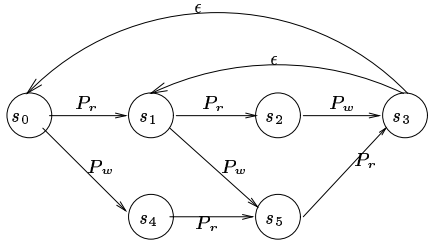

Fig. 1. An example of a scheduler

In this paper, we present a new algorithm solving the linear reachability problem. This algorithm is completely different from the naive one. In our algorithm, we estimate a bound B (called a bounding box) from $\mathbb{A}$ and U such that, the **Question**-part is true iff the truth is witnessed by some p on which the count $\#_{a_i}(p)$ for

each label a_i is bounded by B. Interestingly, after a complex loop analysis, estimating a bounding box B is reduced to a number theory problem: finding nonnegative minimal solutions to linear Diophantine equation systems. There has been much research on this latter problem for homogeneous/inhomogeneous systems with (nonnegative) integer solutions [4,5,13,20]. Suppose that U is in a disjunctive normal form over linear equations/inequalities. Using the Borosh-Flahive-Treybig bound in [4], we are able to show that, in worst cases, when $|S|$ is $\gg$ the size (which will be made clear later in the paper) of U, the bounding box B is bounded by $O(|S|^{k+L+3})$, where L is the maximal number of conjunctions in a single disjunctive term of U. The Borosh-Flahive-Treybig bound has been used in solving the boundedness problem for vector addition systems [22]. However, the path rearrangement technique used in [22] is not applicable (at least not in an easy way) to obtaining the bounding box B. With the bounding box, one can easily show that the linear reachability problem is solvable in time $O(|S|^{2k(k+L+3)+2})$, when $|S| \gg k$ and the size of U. In particular, when k and U are fixed, the complexity is polynomial in $|S|$. This is in contrast to the complexity of the naive algorithm that is exponential in the state number $|S|$. This new complexity result will be further used in this paper to obtain complexity bounds (which were unknown) for some other linear counting problems that involve linear constraints over counts, e.g., the linear liveness problem [11] for $\mathbb{A}$.

We also consider the linear reachability problem when $\mathbb{A}$ is ordered; i.e., on any path p from s_{init} to s_{final}, each label a_j appears after all the a_i's whenever $i < j$. For this restricted model of $\mathbb{A}$, we can obtain a smaller complexity bound $O(|S|^{4k-1})$ for the linear reachability problem, using the Pottier bound [20] (the Borosh-Flahive-Treybig bound is not applicable here) for nonnegative minimal solutions to linear Diophantine systems. Interestingly, this restricted model and the complexity bound can be used to establish a new complexity result for the linear reachability problem for timed automata [1]. The problem is to decide whether there are two tuples of clocks values satisfying two given linear constraints U and U' respectively such that one tuple can reach the other in a timed automaton. For timed automata, although many temporal verification problems involving linear constraints over clocks are known to be undecidable [1,2,12], the linear reachability problem is decidable (even when the clocks are dense) [8–10]. However, an upper bound for the worst-case complexity was unknown. In this paper, we show that the linear reachability problem for a discrete timed automaton (i.e., a timed automaton with integer-valued clocks) is solvable in time $O(|S|^{8k-1})$, when the number of (control) states in the timed automaton, $|S|$, is $\gg$ the number k of clocks, the size of U and U', and the maximal absolute value of all the constants appearing in the clock constraints of the timed automaton. This result can be generalized to timed automata with dense clocks using the pattern technique in [9].

The rest of this paper is organized as follows. Section 2 introduces some known results on minimal solutions to linear Diophantine equation systems, which are needed later in the paper. Section 3 obtains a bounding box for the linear reachabil-

ity problem for finite state transition systems. Based on the bounding box, Section 4 establishes a time complexity bound for the linear liveness problem for finite state transition systems. Section 5 considers ordered finite state transition systems, which, in Section 6, are used to show a time complexity bound for the linear reachability problem for timed automata. Section 7 is a brief conclusion.

2 Preliminaries

Let $\mathbf{N}$ be the set of nonnegative integers and k be a positive integer. A *finite state transition system* $\mathbb{A}$ is defined as

$$\mathbb{A} = \langle S, \Sigma, E \rangle, \tag{1}$$

where S is a finite set of *states*, $\Sigma = \{a_1, \ldots, a_k\}$ is a set of *labels*, $E \subseteq S \times (\Sigma \cup \{\epsilon\}) \times S$ is a set of *transitions*. When $E \subseteq S \times \{\epsilon\} \times S$, $\mathbb{A}$ is called a finite state machine. A *path* p of $\mathbb{A}$ is a finite sequence $(s_0, \tau_0, s_1) \ldots (s_i, \tau_i, s_{i+1}) \ldots (s_{n-1}, \tau_{n-1}, s_n)$ for some n such that for each $0 \leq i < n$, $(s_i, \tau_i, s_{i+1}) \in E$. Path p is a *simple cycle* if $s_0, \ldots, s_{n-1}$ are distinct and $s_0 = s_n$. Path p is a *simple path* if $s_0, \ldots, s_{n-1}, s_n$ are all distinct. For any path p of $\mathbb{A}$, let $\#(p)$ denote the k-ary vector $(\#_{a_1}(p), \ldots, \#_{a_k}(p))$, where each $\#_{a_i}(p)$ stands for the number of label a_i's occurrences on p, $1 \leq i \leq k$.

Let $x_1, \ldots, x_k$ be nonnegative integer variables. An *atomic linear constraint* is in the form of $b_1 x_1 + \ldots + b_k x_k \sim b$, where $\sim \in \{=, \geq\}$, $b_1, \ldots, b_k$ and b are integers. When $\sim$ is $=$ (resp. $\geq$), the constraint is called an *equation* (resp. *inequality*). The constraint is *made homogeneous* if one makes $b = 0$ in the constraint. A *linear constraint* U is a Boolean combination of atomic linear constraints (using $\wedge, \vee, \neg, \rightarrow$). Without loss of generality, throughout this paper, we assume that the linear constraint U is always written as a disjunction $U_1 \vee \ldots \vee U_m$, for some m, of conjunctions of atomic linear constraints. When $m = 1$, U is called a *conjunctive* linear constraint. U is *made homogeneous* if each atomic linear constraint in U is made homogeneous; we use U^{hom} to denote the result. In particular, a conjunctive linear constraint U is a *linear Diophantine equation system* if each atomic linear constraint in U is an equation.

Suppose that U is a conjunctive linear constraint, which contains e equations and $l - e$ inequalities. One may write U into $\mathbf{B}\mathbf{x} \sim \mathbf{b}$, where $\sim \in \{=, \geq\}^l$, $\mathbf{B}$ (l by k) and $\mathbf{b}$ (l by 1) are matrices of integers, and $\mathbf{x}$ is the column of variables $x_1, \ldots, x_k$. As usual, $(\mathbf{B}, \mathbf{b})$ is called the augmented matrix of U, and $\mathbf{B}$ is called the coefficient matrix of U. We use $||\mathbf{B}||_{1,\infty}$ to denote $\max_i\{\sum_j |b_{ij}|\}$ (b_{ij} is the element at row i and column j in $\mathbf{B}$) and use $||\mathbf{b}||_\infty$ to denote the maximum of the absolute values of all the elements in $\mathbf{b}$. Assume r is the rank of $(\mathbf{B}, \mathbf{b})$, and Γ_1 (resp. Γ_2) is the maximum of the absolute values of all the $r \times r$ minors of $\mathbf{B}$ (resp. $(\mathbf{B}, \mathbf{b})$).

When U is a linear Diophantine equation system (i.e., $e = l$), for any given tuples $(v_1, \ldots, v_k)$ and $(v'_1, \ldots, v'_k)$ in $\mathbf{N}^k$, we say $(v_1, \ldots, v_k) \leq (v'_1, \ldots, v'_k)$ if $v_i \leq v'_i$ for all $1 \leq i \leq k$. We say $(v_1, \ldots, v_k) < (v'_1, \ldots, v'_k)$ if $(v_1, \ldots, v_k) \leq (v'_1, \ldots, v'_k)$ and $v_i < v'_i$ for some $1 \leq i \leq k$. A tuple $(v'_1, \ldots, v'_k)$ is a *minimal solution* to U if $(v'_1, \ldots, v'_k)$ is a solution to U but any $(v_1, \ldots, v_k)$ with $(0, \ldots, 0) < (v_1, \ldots, v_k) < (v'_1, \ldots, v'_k)$ is not. Clearly, there are only finitely many minimal solutions to U. It has been an active research area to estimate a bound for minimal solutions, and the following Borosh-Flahive-Treybig bound [4] is needed in this paper.

Theorem 2.1 *(Borosh-Flahive-Treybig bound) A linear Diophantine equation system U has solutions in nonnegative integers iff it has a solution $(x_1, \ldots, x_k)$ in nonnegative integers, such that r unknowns are bounded by Γ_1 and $k-r$ unknowns are bounded by $(\max(k, l) - r + 1)\Gamma_2$.*

The Borosh-Flahive-Treybig bound gives a bound for *one* of the minimal solutions in nonnegative integers to the inhomogeneous system U. In contrast, the following Pottier bound gives an upper bound for *all* of the "minimal solutions" to a conjunctive U (which is not necessarily a linear equation system); this result can be simply obtained from Corollary 1 in [20].

Theorem 2.2 *(Pottier bound) For any conjunctive linear constraint U that contains e equations and $l - e$ inequalities, there are two finite sets S and $S^{\hom} = \{\mathbf{v}_1, \ldots, \mathbf{v}_q\}$, for some q, of vectors in $\mathbf{N}^k$ such that*

- *each element in S (resp. $S^{\hom}$) is a solution to U (resp. $U^{\hom}$),*
- *For any $\mathbf{v} \in \mathbf{N}^k$, $\mathbf{v}$ is a solution to U iff there are $t_1, \ldots, t_q \in \mathbf{N}$, $\mathbf{v} = \mathbf{v}_0 + t_1\mathbf{v}_1 + \ldots + t_q\mathbf{v}_q$ for some $\mathbf{v}_0 \in S$,*
- *each component of all the vectors in $S \cup S^{\hom}$ is bounded by the Pottier bound $(2 + ||\mathbf{B}||_{1,\infty} + ||\mathbf{b}||_\infty)^{k+l+e}$.*

Therefore, for a conjunctive linear constraint U, each of its solutions can be represented as the sum of a small solution and a nonnegative linear combination of small solutions to $U^{\hom}$ (clearly, the inverse is also true). Here, "small" means that the solutions are bounded by the Pottier bound. When U is a linear constraint (i.e., $m \geq 1$), the Pottier bound of U is defined to be the maximal of all the bounds obtained from Theorem 2.2 for each conjunctive linear constraint in U.

An inequality can be translated into an equation by introducing a *slack* variable (e.g., $x_1 - 2x_2 \geq 3$ into $x_1 - 2x_2 - u = 3$, where u, a new variable on $\mathbf{N}$, is the slack variable). So if U is a conjunctive linear constraint (in which there are e equations and $l - e$ inequalities) over $x_1, \ldots, x_k$, we may write U into an equation system $U(x_1, \ldots, x_k, y_1, \ldots, y_{l-e})$ with l equations, where $y_1, \ldots, y_{l-e}$ are the slack variables.

3 A Bounding Box for the Linear Reachability Problem

Let $\mathbb{A}$ be a finite state transition system specified in (1). A set $Q \subseteq \mathbb{N}^k$ is a *small linear set* (with respect to the given $\mathbb{A}$) if Q is in the form of

$$\{\mathbf{e}_0 + \sum_{1 \leq j \leq r} X_j \mathbf{e}_j : \text{ each } X_j \geq 0\}, \tag{2}$$

where nonnegative integer r satisfies $r \leq |S|^k$, k-ary nonnegative integer vectors $\mathbf{e}_0, \ldots, \mathbf{e}_r$ satisfy $||\mathbf{e}_0||_\infty \leq |S|^2$, and for each $j = 1, \ldots, r$, $||\mathbf{e}_j||_\infty \leq |S|$. Q is a *small semilinear set* if it is a union of finitely many small linear sets.

Recall that the linear reachability problem for $\mathbb{A}$ is to decide whether there exists a path p in $\mathbb{A}$ from s_{init} to s_{final} such that p satisfies a given linear constraint $U(x_1, \ldots, x_k)$. Let $\mathbb{P}$ be all paths of $\mathbb{A}$ from s_{init} to s_{final}. We use $\#(\mathbb{P})$ to denote the set of k-ary nonnegative integer vectors $\{\#(p) : p \in \mathbb{P}\}$. Using a complex loop analysis technique to reorganize simple loops on a path, one can show that $\#(\mathbb{P})$ is a small semilinear set.

Lemma 3.1 *$\#(\mathbb{P})$ is a small semilinear set. That is, it can be represented as, for some t,* [2]

$$\#(\mathbb{P}) = \bigcup_{1 \leq i \leq t} Q_i, \tag{3}$$

where each Q_i is a small linear set in the form of (2).

Proof. Let p be a path $(s_0, \tau_0, s_1) \ldots (s_i, \tau_i, s_{i+1}) \ldots (s_{n-1}, \tau_{n-1}, s_n)$ of $\mathbb{A}$. We use $|p| = n$ to denote the length of p, S_p to denote the set of states appearing on p, and p^i to denote the prefix of p whose length is i. Obviously, $|p| \leq |S|$ when p is a simple cycle, and $|p| < |S|$ when p is a simple path. Path p *passes* a state s whenever $s \in S_p$. Given two paths p_1 and p_2, we use $S_{p_1 \cap p_2}$ to denote $S_{p_1} \cap S_{p_2}$, which is the set of all the states that appear on both p_1 and p_2. If $S_{p_1 \cap p_2} \neq \emptyset$, we say that p_1 *touches* p_2 with *touch states* $S_{p_1 \cap p_2}$. Otherwise, we say that p_1 *does not touch* p_2.

Then, we can extract (as shown in **Algorithm 1**), from p, a simple path p_0 (called the *basic path* of p) and a set C_p of simple cycles. It can be observed that the stack content (when reading from bottom to top) does not contain any simple cycles at any moment and hence the basic path p_0 obtained in the last step is indeed a simple path. Define $\Delta_0 = S_{p_0} \cup \{s_0, s_n\}$. In particular, if p_0 is empty, then s_0 must be s_n (i.e., p itself forms a cycle), else $s_0 \in S_{p_0}$ and $s_n \in S_{p_0}$ (i.e., $\Delta_0 = S_{p_0}$). Δ_0 is called the *basic states*.

[2] Note that though t may be large, it is irrelevant here.

Algorithm 1

Initialize a stack ST and a set C_p to be empty;
Scan p from left to right;
for each transition $e = (s_i, \tau_i, s_{i+1})$ on p **do**
  if $s_i = s_{i+1}$ (i.e., e itself is a simple cycle) **then**
    $C_p := C_p \cup \{e\}$;
  else
    Check whether ST, from top to bottom, has an element $e' = (s, \tau, s')$ with $s = s_{i+1}$;
    if yes **then**
      Pop all the elements above e' and e' itself from the stack;
      The popped elements together with e form a simple cycle c;
      $C_p := C_p \cup \{c\}$;
    else
      Push e into ST;
    end if
    p_0 is obtained by concatenating the remaining elements in ST from bottom to top.
  end if
end for

Next, we partition C_p into subsets (called *layers*) $L_1, \ldots, L_m$ for some m as follows. The first layer L_1 is the set of all the simple cycles c in C_p such that c passes a state in Δ_0; i.e., $L_1 = \{c : c \in C_p \text{ and } S_c \cap \Delta_0 \neq \emptyset\}$. Define $\Delta_1 = \cup_{c \in L_1} S_c$ and $T_1 = \cup_{c \in L_1}(S_c \cap \Delta_0) = \Delta_1 \cap \Delta_0$. Δ_1 is the set of all the states that are passed by simple cycles in L_1. T_1 contains exactly all the touch states between p_0 and a simple cycle in L_1. In general, for $i \geq 2$, L_i is the set of all the simple cycles $c \in C_p$ such that c has not been grouped into layers $L_1, \ldots, L_{i-1}$ and c touches some simple cycle in L_{i-1}; i.e., $L_i = \{c : c \in C_p - \cup_{1 \leq j \leq i-1} L_j \text{ and } S_c \cap \Delta_{i-1} \neq \emptyset\}$. Δ_i is the set of all the states that are passed by simple cycles in L_i; i.e., $\Delta_i = \cup_{c \in L_i} S_c$. T_i is the set of all the touch states between a simple cycle in L_{i-1} and a simple cycle in L_i; i.e., $T_i = \cup_{c \in L_i}(S_c \cap \Delta_{i-1}) = \Delta_i \cap \Delta_{i-1}$. It is easy to observe that, according to the above definitions, $L_i \cap L_j = \emptyset$ whenever $i \neq j$, $\Delta_i \cap \Delta_j = \emptyset$ whenever $|i - j| \geq 2$, and $T_i \cap T_j = \emptyset$ whenever $i \neq j$. In particular, since $T_i = \emptyset$ iff $L_i = \emptyset$, $T_i = \emptyset$ implies $T_{i+1} = \emptyset$. Obviously, since each $T_i \subseteq S$, there exists some value $m \leq |S|$ such that $L_1, \ldots, L_m \neq \emptyset$ but $L_{m+1} = \emptyset$. That is, the number of layers is bounded and the bound is independent of the choice of p. We call the tuple $\langle p_0, L_1, \ldots, L_m, T_1, \ldots, T_m \rangle$ the *layered structure* $\mathbb{L}_p$ of path p.

For instance, consider a path p of the transition system in Figure 1 that passes through the states (in this order): $s_0 s_4 s_5 s_3 s_1 s_5 s_3 s_1 s_2 s_3 s_0 s_1 s_5 s_3 s_0 s_4$. After running **Algorithm 1**, we can obtain a basic path $p_0 : s_0 s_4$ and four simple cycles (the labels are omitted for simplicity), $c_1 : s_5 s_3 s_1 s_5$, $c_2 : s_3 s_1 s_2 s_3$, $c_3 : s_0 s_4 s_5 s_3 s_0$, and $c_4 : s_0 s_1 s_5 s_3 s_0$. From the above definitions, they are arranged into two layers as shown in Figure 2. In particular, $T_1 = \{s_0, s_4\}$ and $T_2 = \{s_1, s_3, s_5\}$ are indeed disjoint. Now, suppose that we are given a layered structure $\mathbb{L}_p$, then how can we obtain the path p? Hereafter in this paper, we will use formulas in the form of

Algorithm 2

Initialize $\mathbb{C}$ to be empty;
for each $i = m, \ldots, 2$ **do**
  for each $s \in T_i$ **do**
    Choose an arbitrary simple cycle $c \in L_{i-1}$ that passes s;
    Add c to $\mathbb{C}$.
  end for
end for

$p_0 + \sum_{c \in C_p} X_c c,\ X_c \geq 0$, to stand for those paths obtained from $\mathbb{L}_p$ by traversing the basic path p_0 once, and each simple cycle $c \in C_p$ for X_c times [3] during the traversal of p_0. Obviously, constraints must be put over these X_c's to ensure that we can always obtain a path of the corresponding transition system. For instance, consider the layered structure in Figure 2. In order to obtain p, each of c_i $(i = 1, 2, 3, 4)$ must be traversed at least once (though, for now, we are not interested in the exact numbers of traversals) during the traversal of the basic path p_0. Failing to do so will not allow us to obtain a path; e.g., $p_0 + 2c_1 + 3c_2 + 0c_3 + 0c_4$ corresponds to no path of the transition system in Figure 1 at all. For a layered structure $\mathbb{L}_p$ of a path p of any finite state transition system $\mathbb{A}$, we define Span($\mathbb{L}_p$) as the set of paths obtained by traversing p_0 for once, and traversing each simple cycle in every layer for at least once. That is, Span($\mathbb{L}_p$) is the set $\{q_0 + \sum_{c \in C_p} X_c c : \text{each } X_c \geq 0\}$, where $q_0 = p_0 + \sum_{c \in C_p} c$. Clearly, each path in Span($\mathbb{L}_p$) is indeed a path of $\mathbb{A}$, and $p \in$ Span($\mathbb{L}_p$). Recall that the main objective here is to obtain a small bounding box. However, C_p, the set of simple cycles extracted from p, may be exponentially large (in $|S|$); q_0 may therefore be too long to result in a useful bound. We need to improve the representation of Span($\mathbb{L}_p$) by making q_0 shorter.

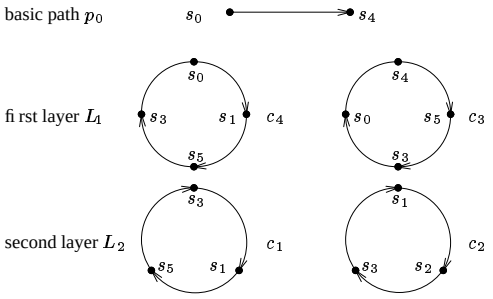

Fig. 2. A layered structure

For each simple cycle c in L_i $(i = 1, \ldots, m)$, it can be observed that $S_c \cap T_i \neq \emptyset$. Also, for each $s \in T_i$ $(i = 2, \ldots, m)$, there exists a simple cycle $c \in L_{i-1}$ that passes s. From these two observations, we can construct a smaller set $\mathbb{C}$ of simple cycles using **Algorithm 2**.

[3] As we are only interested in the counts information of a path, the order in which these cycles should be traversed is irrelevant here.

Obviously $\mathbb{C}$ contains exactly $|T| - |T_1| \leq |S| - |p_0|$ simple cycles, where $T = \bigcup_{1 \leq i \leq m} T_i$, and $q_0' = p_0 + \sum_{c \in \mathbb{C}} c$ constitutes a path of $\mathbb{A}$. Additionally, q_0' has two good properties. One is that the length $|q_0'|$ is bounded by $|p_0| + |\mathbb{C}| \cdot \max_{c \in \mathbb{C}} |c|$. Hence, $|q_0'| \leq |p_0| + (|S| - |p_0|)|S| \leq |S|^2$. Another is that q_0' passes each of the touch states in T, i.e., $T \subseteq S_{q_0'}$. Since each simple cycle $c \in C_p$ passes at least one state in T, we can immediately conclude that $q_0' + \sum_{c \in C_p} X_c c$ constitutes a path of $\mathbb{A}$ for all $X_c \geq 0$. Then, we define Span$'(\mathbb{L}_p)$ as $\{q_0' + \sum_{c \in C_p} X_c c : \text{each } X_c \geq 0\}$. Since $\mathbb{C} \subseteq C_p$, it is easy to see that Span$(\mathbb{L}_p) \subseteq$ Span$'(\mathbb{L}_p)$ and $p \in$Span$'(\mathbb{L}_p)$.

For every p, $\#($Span$'(\mathbb{L}_p)) = \{\#(q_0') + \sum_{c \in C_p} X_c \#(c) : \text{each } X_c \geq 0\}$ is a small linear set. This is because $||\#(q_0')||_\infty \leq |S|^2$, $||\#(c)||_\infty \leq |S|$, and there are at most $r \leq |S|^k$ distinct vectors $\#(c)$ for all simple cycles $c \in C_p$.

Observe that there are only finitely many distinct sets Span$'(\mathbb{L}_p)$ for all $p \in \mathbb{P}$. Since, for each $p \in \mathbb{P}$, Span$'(\mathbb{L}_p) \subseteq \mathbb{P}$, we immediately obtain $\mathbb{P} = \bigcup_{1 \leq i \leq t} \text{Span}'(\mathbb{L}_{p_i})$ for some t and $p_1, \ldots, p_t \in \mathbb{P}$. Define $Q_i = \#(\text{Span}'(\mathbb{L}_{p_i}))$, for $1 \leq i \leq t$. The lemma follows since $\#(\mathbb{P}) = \bigcup_{1 \leq i \leq t} Q_i$ and, as we have shown, each Q_i is a small linear set.

∎

Now let us turn to the property formula U. Recall that U is written as a disjunction of m conjunctive linear constraints

$$U = \bigvee_{1 \leq i \leq m} U_i. \tag{4}$$

Fix any $1 \leq i \leq m$. Suppose that U_i contains l atomic linear constraints. After adding (at most l) slack variables $y_1, \ldots, y_l$, U_i can be written into the following form:

$$\begin{cases} b_{11}x_1 + \ldots + b_{1k}x_k + g_1 y_1 = b_1 \\ \qquad \vdots \\ b_{l1}x_1 + \ldots + b_{lk}x_k + g_l y_l = b_l, \end{cases} \tag{5}$$

where the b's and g's are integers (each g is -1 or 0). Let $\mathbf{B}$ be the coefficient matrix for variables $x_1, \ldots, x_k$ and $\mathbf{b}$ be the column of $b_1, \ldots, b_l$ in (5). Define $w_1 = ||\mathbf{B}||_{1,\infty}$ and $w_2 = ||\mathbf{b}||_\infty$. We may assume $w_1 > 0$ (otherwise let $w_1 = 1$). In the sequel, we use the following notions: W_1 (the maximum of all the values w_1 among all U_i's), W_2 (the maximum of all the values w_2 among all U_i's), and L (the maximum of all the values l among all U_i's).

Due to the disjunctive representations of (4) and (3), we can consider only one conjunction of U in the form of (5) and only one linear set in the form of (2). That is, by substituting the expression in (2) for $\mathbf{x} = (x_1, \ldots, x_k)$ in (5): $\mathbf{x} = \mathbf{e}_0 + \sum_{1 \leq j \leq r} X_j \mathbf{e}_j$, the equation system (5) is transformed into the following equation system with unknowns $X_1, \ldots, X_r$ and $y_1, \ldots, y_l$:

$$\begin{cases} h_{11}X_1 + \ldots + h_{1r}X_r + g_1 y_1 \ = \ d'_1 \\ \qquad \vdots \\ h_{l1}X_1 + \ldots + h_{lr}X_r + g_l y_l \ = \ d'_l. \end{cases} \tag{6}$$

Hence, the linear reachability problem is reduced to finding a nonnegative integer solution to (6). With the bounds on $\mathbf{e}_0$ and each $\mathbf{e}_j$ given in (2), a simple calculation reveals that, in (6), all of the h's are bounded by $|S|W_1$ and all of the $d'_1, \ldots, d'_l$ are bounded by $|S|^2 W_1 + W_2$.

We use Γ_1 to denote the maximum of the absolute values of all the $t \times t$, $1 \leq t \leq l$, minors of the coefficient matrix for system (6) and Γ_2 to denote that of the augmented matrix. With the above mentioned bounds for the coefficients and constants in (6), one can conclude that

$$\Gamma_1 \leq (|S|W_1)^l l! \text{ and } \Gamma_2 \leq (|S|^2 W_1 + W_2)(|S|W_1)^{l-1} l!. \tag{7}$$

A direct application of the Borosh-Flahive-Treybig bound in Theorem 2.1 shows that system (6) has solutions in nonnegative integers iff the system has a solution $(X_1, \ldots, X_r, y_1, \ldots, y_l)$ in nonnegative integers, among which r unknowns are bounded by Γ_1 and l unknowns are bounded by $(r+1)\Gamma_2$ (here, without loss of generality, we assume the worst case where the rank of coefficient matrix of (6) is l). Applying the bounds Γ_1 and $(r+1)\Gamma_2$ to X_j in (2), the linear reachability problem is further reduced to the problem of finding a path $p \in \mathbb{P}$ satisfying:

$$||\#(p)||_\infty \leq (|S|^2 + (r-l)|S|\Gamma_1 + l|S|(r+1)\Gamma_2) \tag{8}$$

and $U(\#_{a_1}(p), \ldots, \#_{a_k}(p))$. Noticing that $l \leq L$, and $r \leq |S|^k$ according to (2), we apply the bounds of Γ_1 and Γ_2 in (7) to (8) and define a bounding box

$$B = (|S|^{k+2} W_1 + L|S|(|S|^k + 1)(|S|^2 W_1 + W_2))(|S|W_1)^{L-1} L! + |S|^2. \tag{9}$$

Hence,

Theorem 3.2 *Given a finite state transition system $\mathbb{A}$, two states $s_{\text{init}}, s_{\text{final}} \in S$, and a linear constraint $U(x_1, \ldots, x_k)$, the following two items are equivalent:*

- *There is a path p of $\mathbb{A}$ from s_{init} to s_{final} satisfying U,*
- *The above item is true for some p further satisfying $||\#(p)||_\infty \leq B$, where B is defined in (9).*

Notice that B in (9) is independent of m in (4). Now we measure the "size" of U with $\max(W_1, W_2, L)$. When the number of states $|S|$ in $\mathbb{A}$ is $\gg k$ and the size of U, the bounding box is in the order of $B = O(|S|^{k+L+3})$. In this case, one can easily show the following.

Theorem 3.3 *The linear reachability problem for finite state transition systems is solvable in time*

$$O(|S|^{2k(k+L+3)+2}), \tag{10}$$

when $|S| \gg k, W_1, W_2, L$.

4 The Linear Liveness Problem

An ω-path π of $\mathbb{A}$ is an infinite sequence such that each prefix is a path of $\mathbb{A}$. Let s and s' be any two designated states of $\mathbb{A}$. We say that π is U-i.o. (infinitely often) at s' if there are infinitely many prefixes p from s to s' of π such that p satisfies U (i.e., $U(\#_{a_1}(p), \ldots, \#_{a_k}(p))$ holds). The *linear liveness problem for finite state transition systems* can be formulated as follows:

- **Given:** A finite state transition system $\mathbb{A}$, two designated states s and s', and a linear constraint $U(x_1, \ldots, x_k)$.
- **Question:** Is there an ω-path π that starts from s and is U-i.o. at s'?

In [11], this problem is shown decidable. However, the time complexity was unknown. In this section, we reduce the liveness problem to a linear reachability problem.

Recall that U is in the form of (4), $U = \vee_{1 \leq i \leq m} U_i$, and U_i^{hom} is the result of making U_i homogeneous. One key observation is as follows. The **Question**-part in the linear liveness problem is true iff, for some $1 \leq i \leq m$, *(a).* there is a path of $\mathbb{A}$ from s to s' satisfying U_i, and, *(b).* there is a path of $\mathbb{A}$ from s' to s' satisfying U_i^{hom}. A proof of this observation can be followed from [11] using the pigeonhole principle. Both items are equivalent to the linear reachability problem for $\mathbb{A}$ concerning U_i and U_i^{hom}, respectively. By trying out all of the m number of U_i's and U_i^{hom}'s, and using Theorem 3.2 and (10), we conclude that:

Theorem 4.1 *The linear liveness problem for finite state transition systems is solvable in time shown in (10), when $|S| \gg m, k, W_1, W_2, L$.*

5 Ordered Finite State Transition Systems

Let $\mathbb{A}$ be a finite state transition system specified in (1). Suppose that an order of labels $a_1, \ldots, a_k$ is fixed, say $a_1 < \ldots < a_k$. $\mathbb{A}$ is *ordered* if, on any path p from s_{init} to s_{final}, each label a_i appears before each label a_j whenever $i < j$. In this case, $\mathbb{A}$ behaves as follows: reading a_1's for 0 or more times, then reading a_2's for 0 or more times, and so on. For this restricted version of $\mathbb{A}$, we can obtain a better complexity bound than (10) for the linear reachability problem.

Lemma 5.1 *The linear reachability problem for ordered $\mathbb{A}$ is solvable in time*

$$O(m \cdot |S|^{4k-2} \cdot P^{2k}), \tag{11}$$

where P is the Pottier bound for U (i.e., the maximum of the Pottier bounds for all U_i's in (4)). Furthermore, since P is independent of $|S|$, the linear reachability problem for ordered $\mathbb{A}$ is solvable in time $O(|S|^{4k-1})$, when $|S| \gg m, k, P$.

Proof. We first assume that U itself is a conjunctive linear constraint (i.e., in (4), $m = 1$) in the form of (5) containing e equations and $l - e$ inequalities. We use P to denote the Pottier bound for U obtained from Theorem 2.2. Obviously, P depends only on U (independent of $|S|$ in $\mathbb{A}$) and is greater than or equal to the Pottier bound for the homogeneous version U^{hom}. In the following, we construct a (finite state) machine M and reduce the linear reachability problem concerning $\mathbb{A}$ and U to a reachability problem for M. Intuitively, $\mathbb{A}$ is a sequential composition of $\mathbb{A}_1, \ldots, \mathbb{A}_k$; each $\mathbb{A}_i$ is a restricted version of $\mathbb{A}$ that reads ϵ and label a_i only. $\mathbb{A}$ can also be treated as $\mathbb{A}_1, \ldots, \mathbb{A}_k$ running concurrently (since they read different labels), as long as each $\mathbb{A}_i$ ends with the state that $\mathbb{A}_{i+1}$ starts with, for each $1 \leq i < k$. In the construction, M simulates these concurrent runs and uses counters c_i to count the number of labels a_i read by $\mathbb{A}_i$, $1 \leq i \leq k$. The key idea is to use Theorem 2.2 to make these counters bounded by the Pottier bound: all these counters are reset to 0 whenever $U^{\text{hom}}(c_1, \ldots, c_k)$ holds.

The finite state machine M works as follows. M is equipped with a tuple variable $\mathbf{s}$ taking values in S^k and k bounded (by the Pottier bound P) nonnegative integer counters $c_1, \ldots, c_k$. Initially, all the counters are 0. M first guesses and remembers $k-1$ states, $s_2^0, \ldots, s_k^0$ in $\mathbb{A}$. Then M sets $\mathbf{s}$ to be $(s_{\text{init}}, s_2^0, \ldots, s_k^0)$. An execution of M consists of some *homogeneous rounds* followed by one *inhomogeneous round*. In a homogeneous round, for each $1 \leq i \leq k$, M executes 0 or more *i-moves*. For each i-move, M updates the i-th component s of $\mathbf{s}$ to s' whenever $\mathbb{A}$ has a transition from s to s' on which the label is a_i or ϵ. Additionally, M increments the counter c_i by 1 whenever the label is a_i. Nondeterministically at some moment, M decides to end this homogeneous round. At this moment, a test of $U^{\text{hom}}(c_1, \ldots, c_k)$ is performed, and when the test is true, M resets every counter to 0. Notice that M crashes whenever one of the counters exceeds the Pottier bound P during the round,

or the test is false. After 0 or more homogeneous rounds, nondeterministically, M decides to start the inhomogeneous round. The inhomogeneous round is exactly as a homogeneous round except that the test is for $U(c_1, \ldots, c_k)$. M *terminates* if, on completing the inhomogeneous round, $\mathbf{s}$ stores $(s_2^0, \ldots, s_k^0, s_{\text{final}})$, where $s_2^0, \ldots, s_k^0$ were guessed and remembered initially. Clearly, because of Theorem 2.2, M terminates iff the ordered $\mathbb{A}$ has a path from s_{init} to s_{final} satisfying U. M is a finite state machine whose state space size is $|S|^{2k-1} \cdot P^k$. Hence, using a depth first search on the graph of M, whether M terminates can be solved in time quadratic to the state space size.

When $m > 1$, one can try each U_i in (4) one by one for the linear reachability problem. The lemma follows. ∎

Recalling the definition of the Borosh-Flahive-Treybig bound, one may notice that it can not be used in the above construction to prove the lemma. In the next section, we will use this restricted model of $\mathbb{A}$ and the complexity bound to study timed automata.

6 The Linear Reachability Problem for Timed Automata

A timed automaton [1] is a finite state machine augmented with a number of clocks. All the clocks progress synchronously with rate 1, except when a clock is reset to 0 at some transition. We first consider discrete timed automata where clocks take integral values. Formally, a *discrete timed automaton* $\mathbb{D}$ is a tuple $\langle S, \{x_1, \ldots, x_k\}, E\rangle$, where S is a finite set of *(control) states*, $x_1, \ldots, x_k$ are *clocks* taking values in $\mathbf{N}$, and E is a finite set of *edges* or *transitions*. Each edge $\langle s, \lambda, l, s'\rangle$ denotes a transition from state s to state s' with *enabling condition* l in the form of clock regions (i.e., $x \# c, x - y \# c$, where x, y are clocks, $\#$ denotes $\leq, \geq$, or $=$, and c is an integer) and a clock reset set $\lambda \subseteq \{1, \ldots, k\}$. Sometimes, we also write the edge as $s \rightarrow_\lambda^l s'$, or simply $s \rightarrow_\lambda s'$ when l is $true$. Without loss of generality, we assume that $|\lambda| \leq 1$. That is, each transition resets at most one clock (since resetting several clocks can be simulated by resetting one by one). When $\lambda = \emptyset$, the edge is called a progress transition. Otherwise, it is a reset transition. $\mathbb{D}$ is *static* if the enabling condition on each edge is simply $true$.

The semantics of $\mathbb{D}$ is defined as follows. A *configuration* of $\mathbb{D}$ is a tuple of a control state and clock values. Let $\langle s, v_1, \ldots, v_k\rangle$ and $\langle s', v_1', \ldots, v_k'\rangle$ be two configurations. $\langle s, v_1, \ldots, v_k\rangle \rightarrow \langle s', v_1', \ldots, v_k'\rangle$ denotes a *one-step transition* satisfying all of the following conditions:

- There is an edge $\langle s, \lambda, l, s'\rangle$ in $\mathbb{D}$,
- The enabling condition of the edge is satisfied; i.e., $l(v_1, \ldots, v_k)$ is true,
- If $\lambda = \emptyset$ (i.e., a progress transition), then every clock progresses by one time

unit; i.e., $v'_i = v_i + 1$, $1 \leq i \leq k$. If, for some j, $\lambda = \{j\}$ (i.e., a reset transition), then x_j resets to 0 and all the other clocks do not change; i.e., $v'_j = 0$ and $v'_i = v_i$ for each $1 \leq i \neq j \leq k$.

$\langle s, v_1, \ldots, v_k \rangle$ *reaches* $\langle s', v'_1, \ldots, v'_k \rangle$ if $\langle s, v_1, \ldots, v_k \rangle \rightarrow^* \langle s', v'_1, \ldots, v'_k \rangle$, where $\rightarrow^*$ is the transitive closure of $\rightarrow$.

The *linear reachability problem for discrete timed automata* is defined as follows.

- **Given:** A discrete timed automaton $\mathbb{D}$, two designated states s_{init} and s_{final}, and two linear constraints U and U' over k variables.
- **Question:** are there clock values $v_1, \ldots, v_k, v'_1, \ldots, v'_k$ such that configuration $\langle s_{\text{init}}, v_1, \ldots, v_k \rangle$ reaches configuration $\langle s_{\text{final}}, v'_1, \ldots, v'_k \rangle$ and both $U(v_1, \ldots, v_k)$ and $U'(v'_1, \ldots, v'_k)$ hold?

It is known that the problem is decidable, even when the clocks are dense. The decidability proofs and application examples can be found in [8–10]. However, as we mentioned earlier, the time complexity for the problem was unknown. And in this section, we give such a complexity bound using the result in (11).

Without loss of generality, we assume that both U and U' in the linear reachability problem for discrete timed automata are disjunctions of m conjunctive linear constraints. Each conjunctive linear constraint contains at most L atomic linear constraints among which there are at most E equations. Similar to Section 3, we use W_1 (resp. W_2) to represent the maximal value of $||\mathbf{B}||_{1,\infty}$ (resp. $||\mathbf{b}||_\infty$) among all the conjunctive linear constraints $\mathbf{Bx} \sim \mathbf{b}$ in U and U'. The complexity of the linear reachability problem will be measured on, among others, L, E, m, W_1, W_2, $|S|$, and k.

We first consider a simpler case when $\mathbb{D}$ is static. Before we proceed further, more definitions are needed. A *reset order* τ is a sequence $\lambda_1, \ldots, \lambda_n$, for some $1 \leq n \leq k$, where each λ_i contains exactly one element in $\{1, \ldots, k\}$, and all of the λ_i's are pair-wisely disjoint. Let $\lambda_0 = \{1, \ldots, k\} - \cup_{1 \leq i \leq n} \lambda_i$. An execution path of $\mathbb{D}$ is *of reset order* τ if every clock x_j with $j \in \lambda_0$ does not reset on p, and for rest of the clocks, their last resets are in this order: $x_{i_1}, \ldots, x_{i_n}$, with $\lambda_1 = \{i_1\}, \ldots, \lambda_n = \{i_n\}$. For the instance of the linear reachability problem of the static $\mathbb{D}$, we consider the **Question**-part witnessed by an execution path that is of any fixed reset order τ (there are only finitely many reset orders). From this instance and the given τ, we will construct an ordered finite state transition system $\mathbb{A}^\tau$ and a linear constraint U^τ. Then, we reduce the linear reachability problem of $\mathbb{D}$ to the linear reachability problem of $\mathbb{A}^\tau$ and obtain the following result.

Lemma 6.1 *The linear reachability problem for static discrete timed automata* $\mathbb{D}$ *is solvable in time*

$$O(k! \cdot m^2 \cdot (k + (k+1) \cdot |S|)^{8k-2} \cdot (2 + k \cdot W_1 + W_2)^{(2k+2L+2E) \cdot 4k}). \qquad (12)$$

Proof. Let τ be any fixed reset order $\lambda_1, \ldots, \lambda_n$. Before we illustrate the construction of $\mathbb{A}^\tau$, some more definitions are needed.

Let $t_{\text{init}} = t_1, \ldots, t_k$ be some given states with $t_1, \ldots, t_{k-1} \notin S$ and $t_k = s_{\text{init}}$. $Add(\mathbb{D})$ is an ordered finite state transition system that repeatedly reads labels b_1 at state t_1 (for 0 or more times nondeterministically), then, labels b_2 at state t_2, ..., labels b_k at state $t_k = s_{\text{init}}$. $Add(\mathbb{D})$ will be used later to "generate" any starting clock values $z_1, \ldots, z_k$ mentioned earlier (each value z_i represents the number of labels b_i read). $Finite(\mathbb{D})$ is a finite state machine in which all the state transitions in $\mathbb{D}$ are kept but clock progresses/resets are ignored. $Reset(\mathbb{D}, \lambda)$ is a finite state machine in which only the state transitions in $\mathbb{D}$ that reset some clock in λ are kept. $Mono(\mathbb{D}, \lambda, a)$ is a finite state transition system (on alphabet $\{a\}$) such that only the state transitions in $\mathbb{D}$ that do not reset a clock in λ are kept. In the meantime, $Mono(\mathbb{D}, \lambda, a)$ replaces every progress transition in $\mathbb{D}$ by a transition with label a. These finite state transition systems will be used as basic "building blocks" in constructing $\mathbb{A}^\tau$.

To construct $\mathbb{A}^\tau$, we have two cases to consider. The first case is when $\lambda_0 \neq \emptyset$ (i.e., $n < k$). In this case, an execution path of $\mathbb{D}$ can be partitioned into $n+1$ segments separated by the n last resets given in τ. We use $y_0, y_1, \ldots, y_n$ to denote the number of progress transitions made on each segment respectively. Suppose that the path starts with clock values $z_1, \ldots, z_k$ and ends with clock values $x_1, \ldots, x_k$. Corresponding to the $n+1$ segments, we construct $\mathbb{A}^\tau$ to be a sequential composition of $n+1$ finite state transition systems (in this order): $M_0, \ldots, M_n$, where M_0 is the sequential composition of $Add(\mathbb{D})$ and $Mono(\mathbb{D}, \lambda_0, a_0)$, and each M_i, $1 \leq i \leq n$, is the sequential composition of one move in $Reset(\mathbb{D}, \lambda_i)$ and then $Mono(\mathbb{D}, \cup_{0 \leq j \leq i} \lambda_j, a_i)$. Clearly, $\mathbb{A}^\tau$ has alphabet $\{b_1, \ldots, b_k, a_0, \ldots, a_n\}$ with $n < k$ and $\mathbb{A}^\tau$ is ordered (with the ordering of labels $b_1 < \ldots b_k < a_0 < \ldots < a_n$). On a path p of $\mathbb{A}^\tau$, the counts $\#_{b_1}(p), \ldots, \#_{b_k}(p), \#_{a_0}(p), \ldots, \#_{a_n}(p)$ correspond to the above mentioned values $z_1, \ldots, z_k, y_0, \ldots, y_n$, respectively. Each ending clock value x_i can be represented as a summation of (some of) $z_1, \ldots, z_k$ and $y_0, y_1, \ldots, y_n$. More precisely, for each $1 \leq i \leq k$, we use $h(i)$ to denote the number $0 \leq j \leq n$ with $i \in \lambda_j$. Then, for each $i \in \lambda_0$,

$$x_i = z_i + \sum_{h(i) \leq j \leq n} y_j, \tag{13}$$

and for each $i \notin \lambda_0$,

$$x_i = \sum_{h(i) \leq j \leq n} y_j. \tag{14}$$

Substituting (13) and (14) for each x_i, $1 \leq i \leq k$, in $U'(x_1, \ldots, x_k)$, one can obtain

a linear constraint R over $z_1, \ldots, z_k, y_0, \ldots, y_n$. We use Q^τ to denote

$$R(z_1, \ldots, z_k, y_0, \ldots, y_n) \wedge U(z_1, \ldots, z_k). \tag{15}$$

The second case is when $\lambda_0 = \emptyset$ (i.e., $n = k$). In this case, we only need to replace M_0 in the above construction for $\mathbb{A}^\tau$ with $Finite(\mathbb{D})$. The resulting $\mathbb{A}^\tau$ is then an ordered finite state transition system over alphabet $\{b_1, \ldots, b_k, a_1, \ldots, a_k\}$. In this second case, for each $1 \leq i \leq k$, we use $h(i)$ to denote the number $1 \leq j \leq n$ with $i \in \lambda_j$. Similarly, one may obtain, from $U'(x_1, \ldots, x_k)$, a linear constraint R over $y_1, \ldots, y_k$ using the following substitutions:

$$x_i = \sum_{h(i) \leq j \leq k} y_j. \tag{16}$$

In this case, we use Q^τ to denote

$$R(y_1, \ldots, y_k) \wedge U(z_1, \ldots, z_k). \tag{17}$$

In both cases, the number of states in $\mathbb{A}^\tau$ is at most $k + (k+1) \cdot |S|$. It is straightforward to verify the following claim:

(*) The **Question**-part for the linear reachability problem for static $\mathbb{D}$ is true iff, for some τ, there is a path p from t_{init} (the initial state of machine $Add(\mathbb{D})$) to s_{final} in ordered $\mathbb{A}^\tau$ such that p satisfies Q'.

Notice that Q^τ does not depend on $|S|$. In order to use (11) on finite state transition system $\mathbb{A}^\tau$, we will estimate the Pottier bound for Q^τ as follows. Q^τ, in the form of (15) or (17), contains at most $2k$ variables. Both of the forms can be reorganized into a disjunctive normal form. That is, Q^τ can be written into at most m^2 conjunctive linear constraints Q_j. Each Q_j is a conjunction of a conjunctive linear constraint in U' (using substitutions like (13,14,16)) and a conjunctive linear constraint in U. It is easy to see that the Pottier bound, using Theorem 2.2, of Q_j is at most $(2 + k \cdot W_1 + W_2)^{2k+2L+2E}$. Hence, the Pottier bound for Q^τ is also bounded by the same number. Notice that there are at most $3 \cdot k!$ distinct τ's. So, using (11) and the above claim, the linear reachability problem for static $\mathbb{D}$ is solvable in time shown in (12). ∎

We use $\max(m, L, W_1, W_2)$ to measure the "size" of U and U'. Using (12) and noticing that $E \leq L$, we further conclude that the linear reachability problem for static $\mathbb{D}$ is solvable in time $O(|S|^{8k-1})$, when $|S| \gg k$ and the size of U and U'.

Next, we consider the case when $\mathbb{D}$ is not necessarily static. Let C be one plus the maximal absolute value of all the constants appearing in enabling conditions in $\mathbb{D}$. We use T to denote the result of (12) after replacing $|S|$ with $(1 + 2C)^{k^2+k} \cdot |S|$, L with $L + k$, E with $E + k$, W_1 with $\max(W_1, 2)$, and W_2 with $\max(W_2, C)$.

Theorem 6.2 *The linear reachability problem for discrete timed automata $\mathbb{D}$ is solvable in time $O(k! \cdot (1+C)^k \cdot T)$.*

Proof. Let R be a linear constraint in the following form:

$$x_{i_1} \sim_1 c_1 \wedge x_{i_2} - x_{i_1} \sim_2 c_2 \wedge \ldots \wedge x_{i_k} - x_{i_{k-1}} \sim_k c_k, \tag{18}$$

where $(i_1, \ldots, i_k)$ is a permutation of $(1, \ldots, k)$, each $\sim_j$ is "=" iff $0 \leq c_j < C$, and each $\sim_j$ is "$\geq$" iff $c_j = C$. Let R be fixed. From [12,10], one can construct a static $\mathbb{D}'$ with two designated states s'_{init} and s'_{final} and with at most $(1+2C)^{k^2+k} \cdot |S|$ number of states to simulate $\mathbb{D}$ faithfully. More precisely, $\mathbb{D}'$ has this nice property: for any clock values $v_1, \ldots, v_k, v'_1, \ldots, v'_k \in \mathbf{N}$ satisfying $R(v_1, \ldots, v_k)$, the following two items are equivalent,

(I). $\langle s_{\text{init}}, v_1, \ldots, v_k \rangle$ reaches $\langle s_{\text{final}}, v'_1, \ldots, v'_k \rangle$ in $\mathbb{D}$,

(II). $\langle s'_{\text{init}}, v_1, \ldots, v_k \rangle$ reaches $\langle s'_{\text{final}}, v'_1, \ldots, v'_k \rangle$ in $\mathbb{D}'$.

Hence, to solve the linear reachability problem for $\mathbb{D}$, one needs only to solve, for each choice of R, the linear reachability problem for $\mathbb{D}'$:

> (**) Are there clock values $v_1, \ldots, v_k, v'_1, \ldots, v'_k \in \mathbf{N}$ satisfying the following two conditions: $\langle s'_{\text{init}}, v_1, \ldots, v_k \rangle$ reaches $\langle s'_{\text{final}}, v'_1, \ldots, v'_k \rangle$ in $\mathbb{D}'$ and both $U(v_1, \ldots, v_k) \wedge R(v_1, \ldots, v_k)$ and $U'(v'_1, \ldots, v'_k)$ hold?

With the representation of R in (18), the linear reachability problem for static $\mathbb{D}'$ shown in (**) can be solved in time as in (12), after replacing $|S|$ with $(1+2C)^{k^2+k} \cdot |S|$, L with $L+k$, E with $E+k$, W_1 with $\max(W_1, 2)$, and W_2 with $\max(W_2, C)$. As the total choices for R in (18) are at most $k! \cdot (1+C)^k$, the linear reachability problem for discrete timed automata is solvable in time $O(k! \cdot (1+C)^k \cdot T)$. ∎

From Theorem 6.2 and the definition of T, we can further conclude that the linear reachability problem for discrete timed automata is solvable in time $O(|S|^{8k-1})$, when $|S|$ is $\gg k, C$, and the size of U and U'.

Now, we turn to the case when $\mathbb{D}$ is a timed automaton with k dense clocks. One can similarly formulate the semantics and the linear reachability problem for $\mathbb{D}$ (e.g., see [9]). With the pattern technique presented in [9], it is easy to show the following. From $\mathbb{D}$ and U, U', one can construct a discrete timed automaton $\mathbb{D}'$ with k discrete clocks and two linear constraints W, W' such that the linear reachability problem of timed automaton $\mathbb{D}$ concerning U, U' is equivalent to the linear reachability problem of discrete timed automaton $\mathbb{D}'$ concerning W, W'. In addition, the number of states in $\mathbb{D}'$ is $O(2^{6(k+1)^2} \cdot |S|)$, where S is the state set in $\mathbb{D}$. (There are at most $2^{6(k+1)^2}$ *patterns* [9].) Furthermore, W and W' only depend on U, U' and k (independent of $\mathbb{D}$). Hence, the linear reachability problem for $\mathbb{D}$ with dense clocks is still solvable

in time $O(|S|^{8k-1})$, when $|S| \gg k, C,$ and the size of U and U'. To sum up, we have the following result.

Theorem 6.3 *The linear reachability problem for timed automata (with integer-valued clocks or with dense clocks) is solvable in time $O(|S|^{8k-1})$, when $|S| \gg k, C, m, L, W_1, W_2$.*

7 Conclusions

In this paper, we obtained a number of new complexity results for various linear counting problems (reachability and liveness) for (ordered) finite state transition systems and timed automata. At the heart of the proofs, we used some known results in estimating the upper bound for minimal solutions (in nonnegative integers) for linear Diophantine systems. In particular, when all the parameters (such as the number of labels/clocks, the largest constant C in a timed automaton, the size of the linear constraint to be verified, etc.) except the number of states, $|S|$, of the underlying transition system are considered constants, all of the complexity bounds obtained in this paper is polynomial in $|S|$. This is, as we mentioned in Section 1, in contrast to the exponential bounds that were previously known. In practice, a requirement specification (e.g., the U in a linear counting problem) is usually small and simple [14]. In this sense, our results are useful, since the large $|S|$ is usually the dominant factor in efficiently solving these verification problems. However, in real-world applications, how to use the structural information (such as modularity) of a transition system to obtain a smaller bounding box remains a practical problem to solve.

The authors would like to thank P. San Pietro and the anonymous referees for many valuable suggestions.

References

[1] R. Alur and D. L. Dill. A theory of timed automata. *Theoretical Computer Science*, 126(2):183–235, April 1994.

[2] R. Alur and T. A. Henzinger. A really temporal logic. *Journal of the ACM*, 41(1):181–204, January 1994.

[3] A. Biere, A. Cimatti, E. Clarke, and Y. Zhu. Symbolic model checking without BDDs. In *TACAS'99*, volume 1579 of *LNCS*, pages 193–207. Springer-Verlag, 1999.

[4] I. Borosh, M. Flahive, and B. Treybig. Small solutions of linear diophantine equations. *Discrete Mathematics*, 58:215–220, 1986.

[5] I. Borosh and B. Treybig. Bounds on positive integral solutions of linear diophantine equations. *Proceedings of the American Mathematical Society*, 55:299–304, 1976.

[6] A. Bouajjani, R. Echahed, and P. Habermehl. On the verification problem of nonregular properties for nonregular processes. In *LICS'95*, pages 123–133. IEEE CS Press, 1995.

[7] E. M. Clarke, E. A. Emerson, and A. P. Sistla. Automatic verification of finite-state concurrent systems using temporal logic specifications. *TOPLAS*, 8(2):244–263, 1986.

[8] H. Comon and Y. Jurski. Timed automata and the theory of real numbers. In *CONCUR'99*, volume 1664 of *LNCS*, pages 242–257. Springer, 1999.

[9] Zhe Dang. Pushdown timed automata: a binary reachability characterization and safety verification. *Theoretical Computer Science*, 302(1-3): 93–121, 2003.

[10] Zhe Dang, O. H. Ibarra, T. Bultan, R. A. Kemmerer, and J. Su. Binary reachability analysis of discrete pushdown timed automata. In *CAV'00*, volume 1855 of *LNCS*, pages 69–84. Springer, 2000.

[11] Zhe Dang, O. H. Ibarra, and P. San Pietro. Liveness verification of reversal-bounded multicounter machines with a free counter. In *FSTTCS'01*, volume 2245 of *LNCS*, pages 132–143. Springer, 2001.

[12] Zhe Dang, P. San Pietro, and R. A. Kemmerer. Presburger liveness verification of discrete timed automata. *Theoretical Computer Science*, 299(1-3): 413–438, 2003.

[13] E. Domenjoud. Solving systems of linear diophantine equations: an algebraic approach. In *MFCS'91*, volume 520 of *LNCS*, pages 141–150. Springer-Verlag, 1991.

[14] Matthew B. Dwyer, George S. Avrunin, and James C. Corbett. Patterns in property specifications for finite-state verification. In *ICSE'99*, pages 411–421. ACM Press, 1999.

[15] G. J. Holzmann. The model checker SPIN. *TSE*, 23(5):279–295, May 1997.

[16] J. Hopcroft and J. Ullman. *Introduction to Automata theory, Languages, and Computation*. Addison-Wesley Publishing Company, 1979.

[17] K.L. McMillan. *Symbolic Model Checking*. Kluwer Academic Publishers, Norwell Massachusetts, 1993.

[18] R. Parikh. On context-free languages. *JACM*, 13:570–581, 1966.

[19] A. Pnueli. The temporal logic of programs. In *FOCS'77*, pages 46–57. IEEE CS Press, 1977.

[20] L. Pottier. Minimal solutions of linear diophantine equations: Bounds and algorithms. In *Rewriting Techniques and Applications*, volume 488 of *LNCS*, pages 162–173. Springer-Verlag, 1991.

[21] M. Y. Vardi and P. Wolper. An automata-theoretic approach to automatic program verification (preliminary report). In *LICS'86*, pages 332–344. IEEE CS Press, 1986.

[22] C. Rackoff. The covering and boundedness problems for vector addition systems. *Theoretical Computer Science*, 6:223–231, 1978.