Ranking Templates for Linear Loops*

Jan Leike[1,2] and Matthias Heizmann[1**]

[1] University of Freiburg, Germany
[2] Max Planck Institute for Software Systems

Abstract. We present a new method for the constraint-based synthesis of termination arguments for linear loop programs based on *linear ranking templates*. Linear ranking templates are parametrized, well-founded relations such that an assignment to the parameters gives rise to a ranking function. This approach generalizes existing methods and enables us to use templates for many different ranking functions with affine-linear components. We discuss templates for multiphase, piecewise, and lexicographic ranking functions. Because these ranking templates require both strict and non-strict inequalities, we use Motzkin's Transposition Theorem instead of Farkas Lemma to transform the generated ∃∀-constraint into an ∃-constraint.

1 Introduction

The scope of this work is the constraint-based synthesis of termination arguments. In our setting, we consider *linear loop programs*, which are specified by a boolean combination of affine-linear inequalities over the program variables. This allows for both, deterministic and non-deterministic updates of the program variables. An example of a linear loop program is given in Figure 1.

Usually, linear loop programs do not occur as stand-alone programs. Instead, they are generated as a finite representation of an infinite path in a control flow graph. For example, in (potentially spurious) counterexamples in termination analysis [1,2,3,4,5,6], non-termination analysis [7], stability analysis [8,9], or cost analysis [10,11].

We introduce the notion of *linear ranking templates* (Section 3). These are parameterized relations specified by linear inequalities such that any assignment to the parameters yields a well-founded relation. This notion is general enough to encompass all existing methods for linear loop programs that use constraint-based synthesis of ranking functions of various kinds (see Section 6 for an assessment). Moreover, ours is the first method for synthesis of lexicographic ranking functions that does not require a mapping between loop disjuncts and lexicographic components.

In this paper we present the following linear ranking templates.

* The final publication is available at `http://link.springer.com`.

** This work is supported by the German Research Council (DFG) as part of the Transregional Collaborative Research Center "Automatic Verification and Analysis of Complex Systems" (SFB/TR14 AVACS)

```
while (q > 0):
    q := q - y;
    y := y + 1;
```

$$q > 0 \\ \wedge\, q' = q - y \\ \wedge\, y' = y + 1$$

Fig. 1. A linear loop program given as program code (left) and as a formula defining a binary relation (right).

- The *multiphase ranking template* specifies a ranking function that proceeds through a fixed number of phases in the program execution. Each phase is ranked by an affine-linear function; when this function becomes non-positive, we move on to the next phase (Subsection 4.1).
- The *piecewise ranking template* specifies a ranking function that is a piecewise affine-linear function with affine-linear predicates to discriminate between the pieces (Subsection 4.2).
- The *lexicographic ranking template* specifies a lexicographic ranking function that corresponds to a tuple of affine-linear functions together with a lexicographic ordering on the tuple (Subsection 4.3).

These linear ranking templates can be used as a 'construction kit' for composing linear ranking templates that enable more complex ranking functions (Subsection 4.4). Moreover, variations on the linear ranking templates presented here can be used and completely different templates could be conceived.

Our method is described in Section 5 and can be summarized as follows. The input is a linear loop program as well as a linear ranking template. From these we construct a constraint to the parameters of the template. With Motzkin's Theorem we can transform the constraint into a purely existentially quantified constraint (Subsection 5.1). This $\exists$-constraint is then passed to an SMT solver which checks it for satisfiability. A positive result implies that the program terminates. Furthermore, a satisfying assignment will yield a ranking function, which constitutes a termination argument for the given linear loop program.

Related approaches invoke Farkas' Lemma for the transformation into $\exists$-constraints [12,13,14,15,16,17,18,19]. The piecewise and the lexicographic ranking template contain both strict and non-strict inequalities, yet only non-strict inequalities can be transformed using Farkas' Lemma. We solve this problem by introducing the use of Motzkin's Transposition Theorem, a generalization of Farkas' Lemma. As a side effect, this also enables both, strict and non-strict inequalities in the program syntax. To our knowledge, all of the aforementioned methods can be improved by the application of Motzkin's Theorem instead of Farkas' Lemma.

Our method is complete in the following sense. If there is a ranking function of the form specified by the given linear ranking template, then our method will discover this ranking function. In other words, the existence of a solution is never lost in the process of transforming the constraint.

In contrast to some related methods [16,17] the constraint we generate is not linear, but rather a nonlinear algebraic constraint. Theoretically, this constraint can be decided in exponential time [20]. Much progress on nonlinear SMT solvers has been made and present-day implementations routinely solve nonlinear constraints of various sizes [21].

A related setting to linear loop programs are *linear lasso programs*. These consist of a linear loop program and a program stem, both of which are specified by boolean combinations of affine-linear inequalities over the program variables. Our method can be extended to linear lasso programs through the addition of affine-linear inductive invariants, analogously to related approaches [13,15,16,19].

2 Preliminaries

In this paper we use $\mathbb{K}$ to denote a field that is either the rational numbers $\mathbb{Q}$ or the real numbers $\mathbb{R}$. We use ordinal numbers according to the definition in [22]. The first infinite ordinal is denoted by ω; the finite ordinals coincide with the natural numbers, therefore we will use them interchangeably.

2.1 Motzkin's Transposition Theorem

Intuitively, Motzkin's Transposition Theorem [23, Corollary 7.1k] states that a given system of linear inequalities has no solution if and only if a contradiction can be derived via a positive linear combination of the inequalities.

Theorem 1 (Motzkin's Transposition Theorem). *For $A \in \mathbb{K}^{m\times n}$, $C \in \mathbb{K}^{\ell\times n}$, $b \in \mathbb{K}^m$, and $d \in \mathbb{K}^\ell$, the formulae* (M1) *and* (M2) *are equivalent.*

$$\forall x \in \mathbb{K}^n.\ \neg(Ax \leq b \ \wedge\ Cx < d) \tag{M1}$$

$$\begin{aligned} \exists \lambda \in \mathbb{K}^m\ \exists \mu \in \mathbb{K}^\ell.\ \lambda \geq 0 \ \wedge\ \mu \geq 0 \\ \wedge\ \lambda^T A + \mu^T C = 0 \ \wedge\ \lambda^T b + \mu^T d \leq 0 \\ \wedge\ (\lambda^T b < 0 \ \vee\ \mu \neq 0) \end{aligned} \tag{M2}$$

If ℓ is set to 1 in Theorem 1, we obtain the affine version of Farkas' Lemma [23, Corollary 7.1h]. Therefore Motzkin's Theorem is strictly superior to Farkas' Lemma, as it allows for a combination of both strict and non-strict inequalities. Moreover, it is logically optimal in the sense that it enables transformation of *any* universally quantified formula from the theory of linear arithmetic.

2.2 Linear Loop Programs

In this work, we consider programs that consist of a single loop. We use binary relations over the program's states to define its transition relation.

We denote by x the vector of n variables $(x_1, \ldots, x_n)^T \in \mathbb{K}^n$ corresponding to program states and by $x' = (x'_1, \ldots, x'_n)^T \in \mathbb{K}^n$ the variables of the next state.

Definition 1 (Linear loop program). *A* linear loop program $\textsc{loop}(x, x')$ *is a binary relation defined by a formula with the free variables x and x' of the form*

$$\bigvee_{i \in I} \left(A_i \binom{x}{x'} \leq b_i \ \wedge \ C_i \binom{x}{x'} < d_i \right)$$

for some finite index set I, some matrices $A_i \in \mathbb{K}^{n \times m_i}$, $C_i \in \mathbb{K}^{n \times k_i}$, and some vectors $b_i \in \mathbb{K}^{m_i}$ and $d_i \in \mathbb{K}^{k_i}$. The linear loop program $\textsc{loop}(x, x')$ is called conjunctive*, iff there is only one disjunct, i.e., $\#I = 1$.*

Geometrically the relation LOOP corresponds to a union of convex polyhedra.

Definition 2 (Termination). *We say that a linear loop program $\textsc{loop}(x, x')$* terminates *iff the relation $\textsc{loop}(x, x')$ is well-founded.*

Example 1. Consider the following program code.

```
while (q > 0):
    if (y > 0):
        q := q - y - 1;
    else:
        q := q + y - 1;
```

We represent this as the following linear loop program:

$$\begin{aligned} &(q > 0 \ \wedge \ y > 0 \ \wedge \ y' = y \ \wedge \ q' = q - y - 1) \\ \vee\, &(q > 0 \ \wedge \ y \leq 0 \ \wedge \ y' = y \ \wedge \ q' = q + y - 1) \end{aligned}$$

This linear loop program is not conjunctive. Furthermore, there is no infinite sequence of states $x_0, x_1, \ldots$ such that for all $i \geq 0$, the two successive states (x_i, x_{i+1}) are contained in the relation LOOP. Hence the relation $\textsc{loop}(x, x')$ is well-founded and the linear loop program terminates.

3 Ranking Templates

A *ranking template* is a template for a well-founded relation. More specifically, it is a parametrized formula defining a relation that is well-founded for all assignments to the parameters. If we show that a given program's transition relation LOOP is a subset of an instance of this well-founded relation, it must be well-founded itself and thus we have a proof for the program's termination. Moreover, an assignment to the parameters of the template gives rise to a ranking function. In this work, we consider ranking templates that can be encoded in linear arithmetic.

We call a formula whose free variables contain x and x' a *relation template*. Each free variable other than x and x' in a relation template is called *parameter*. Given an assignment ν to all parameter variables of a relation template $\textsc{t}(x, x')$,

the evaluation $\nu(\textsc{t})$ is called an *instantiation of the relation template* $\textsc{t}$. We note that each instantiation of a relation template $\textsc{t}(x, x')$ defines a binary relation.

When specifying templates, we use parameter variables to define affine-linear functions. For notational convenience, we will write $f(x)$ instead of the term $s_f^T x + t_f$, where $s_f \in \mathbb{K}^n$ and $t_f \in \mathbb{K}$ are parameter variables. We call f an *affine-linear function symbol.*

Definition 3 (Linear ranking template). *Let* $\textsc{t}(x, x')$ *be a template with parameters* D *and affine-linear function symbols* F *that can be written as a boolean combination of atoms of the form*

$$\sum_{f \in F} \big(\alpha_f \cdot f(x) + \beta_f \cdot f(x')\big) + \sum_{d \in D} \gamma_d \cdot d \ \rhd \ 0,$$

where $\alpha_f, \beta_f, \gamma_d \in \mathbb{K}$ *are constants and* $\rhd \in \{\geq, >\}$. *We call* $\textsc{t}$ *a* linear ranking template *over* D *and* F *iff every instantiation of* $\textsc{t}$ *defines a well-founded relation.*

Example 2. We call the following template with parameters $D = \{\delta\}$ and affine-linear function symbols $F = \{f\}$ the *Podelski-Rybalchenko ranking template* [17].

$$\delta > 0 \ \wedge \ f(x) > 0 \ \wedge \ f(x') < f(x) - \delta \tag{1}$$

In the remainder of this section, we introduce a formalism that allows us to show that every instantiation of the Podelski-Rybalchenko ranking template defines a well-founded relation. Let us now check the additional syntactic requirements for 1 to be a linear ranking template:

$$\begin{aligned} \delta > 0 \ &\equiv \ \big(0 \cdot f(x) + 0 \cdot f(x')\big) + 1 \cdot \delta > 0 \\ f(x) > 0 \ &\equiv \ \big(1 \cdot f(x) + 0 \cdot f(x')\big) + 0 \cdot \delta > 0 \\ f(x') < f(x) - \delta \ &\equiv \ \big(1 \cdot f(x) + (-1) \cdot f(x')\big) + (-1) \cdot \delta > 0 \end{aligned}$$

The next lemma states that we can prove termination of a given linear loop program by checking that this program's transition relation is included in an instantiation of a linear ranking template.

Lemma 1. *Let* $\textsc{loop}$ *be a linear loop program and let* $\textsc{t}$ *be a linear ranking template with parameters* D *and affine-linear function symbols* F. *If there is an assignment* ν *to* D *and* F *such that the formula*

$$\forall x, x'. \ \big(\textsc{loop}(x, x') \rightarrow \nu(\textsc{t})(x, x')\big) \tag{2}$$

is valid, then the program $\textsc{loop}$ *terminates.*

Proof. By definition, $\nu(\textsc{t})$ is a well-founded relation and (2) is valid iff the relation $\textsc{loop}$ is a subset of $\nu(\textsc{t})$. Thus $\textsc{loop}$ must be well-founded. $\square$

In order to establish that a formula conforming to the syntactic requirements is indeed a ranking template, we must show the well-foundedness of its instantiations. According to the following lemma, we can do this by showing that an assignment to D and F gives rise to a ranking function. A similar argument was given in [24]; we provide significantly shortened proof by use of the Recursion Theorem, along the lines of [22, Example 6.12].

Definition 4 (Ranking Function). *Given a binary relation R over a set Σ, we call a function ρ from Σ to some ordinal α a* ranking function *for R iff for all $x, x' \in \Sigma$ the following implication holds.*

$$(x, x') \in R \implies \rho(x) > \rho(x')$$

Lemma 2. *A binary relation R is well-founded if and only if there exists a ranking function for R.*

Proof. Let ρ be a ranking function for R. The image of a sequence decreasing with respect to R under ρ is a strictly decreasing ordinal sequence. Because the ordinals are well-ordered, this sequence cannot be infinite.

Conversely, the graph $G = (\Sigma, R)$ with vertices Σ and edges R is acyclic by assumption. Hence the function ρ that assigns to every element of Σ an ordinal number such that $\rho(x) = \sup\{\rho(x') + 1 \mid (x, x') \in R\}$ is well-defined and exists due to the Recursion Theorem [22, Theorem 6.11]. □

Example 3. Consider the terminating linear loop program LOOP from Example 1. A ranking function for LOOP is $\rho : \mathbb{R}^2 \to \omega$, defined as follows.

$$\rho(q, y) = \begin{cases} \lceil q \rceil, & \text{if } q > 0, \text{ and} \\ 0 & \text{otherwise.} \end{cases}$$

Here $\lceil \cdot \rceil$ denotes the ceiling function that assigns to every real number r the smallest natural number that is larger or equal to r. Since we consider the natural numbers to be a subset of the ordinals, the ranking function ρ is well-defined.

We use assignments to a template's parameters and affine-linear function symbols to construct a ranking function. These functions are real-valued and we will transform them into functions with codomain ω in the following way.

Definition 5. *Given an affine-linear function f and a real number $\delta > 0$ called the* step size*, we define the* ordinal ranking equivalent *of f as*

$$\widehat{f}(x) = \begin{cases} \left\lceil \frac{f(x)}{\delta} \right\rceil, & \textit{if } f(x) > 0, \textit{ and} \\ 0 & \textit{otherwise.} \end{cases}$$

For better readability we used this notation which does not explicitly refer to δ. In our presentation the step size δ is always clear from the context in which an ordinal ranking equivalent $\widehat{f}$ is used.

Example 4. Consider the linear loop program $\textsc{loop}(x, x')$ from Example 1. For $\delta = \frac{1}{2}$ and $f(q) = q + 1$, the ordinal ranking equivalent of f with step size δ is

$$\widehat{f}(q, y) = \begin{cases} \lceil 2(q+1) \rceil, & \text{if } q + 1 > 0, \text{ and} \\ 0 & \text{otherwise.} \end{cases}$$

The assignment from Example 4 to δ and f makes the implication (2) valid. In order to invoke Lemma 1 to show that the linear loop program given in Example 1 terminates, we need to prove that the Podelski-Rybalchenko ranking template is a linear ranking template. We use the following technical lemma.

Lemma 3. *Let f be an affine-linear function of step size $\delta > 0$ and let x and x' be two states. If $f(x) > 0$ and $f(x) - f(x') > \delta$, then $\widehat{f}(x) > 0$ and $\widehat{f}(x) > \widehat{f}(x')$.*

Proof. From $f(x) > 0$ follows that $\widehat{f}(x) > 0$. Therefore $\widehat{f}(x) > \widehat{f}(x')$ in the case $\widehat{f}(x') = 0$. For $\widehat{f}(x') > 0$, we use the fact that $f(x) - f(x') > \delta$ to conclude that $\frac{f(x)}{\delta} - \frac{f(x')}{\delta} > 1$ and hence $\widehat{f}(x') > \widehat{f}(x)$. □

An immediate consequence of this lemma is that the Podelski-Rybalchenko ranking template is a linear ranking template: any assignment ν to δ and f satisfies the requirements of Lemma 3. Consequently, $\widehat{f}$ is a ranking function for $\nu(\textsc{t})$, and by Lemma 2 this implies that $\nu(\textsc{t})$ is well-founded.

4 Examples of Ranking Templates

4.1 Multiphase Ranking Template

The multiphase ranking template is targeted at programs that go through a finite number of phases in their execution. Each phase is ranked with an affine-linear function and the phase is considered to be completed once this function becomes non-positive.

Example 5. Consider the linear loop program from Figure 1. Every execution can be partitioned into two phases: first y increases until it is positive and then q decreases until the loop condition $q > 0$ is violated. Depending on the initial values of y and q, either phase might be skipped altogether.

Definition 6 (Multiphase Ranking Template). *We define the k-*phase ranking template *with parameters $D = \{\delta_1, \ldots, \delta_k\}$ and affine-linear function symbols*

$F = \{f_1, \ldots, f_k\}$ *as follows.*

$$\bigwedge_{i=1}^{k} \delta_i > 0 \tag{3}$$

$$\wedge \bigvee_{i=1}^{k} f_i(x) > 0 \tag{4}$$

$$\wedge\ f_1(x') < f_1(x) - \delta_1 \tag{5}$$

$$\wedge \bigwedge_{i=2}^{k} \Big(f_i(x') < f_i(x) - \delta_i \ \vee\ f_{i-1}(x) > 0 \Big) \tag{6}$$

We say that the multiphase ranking function given by an assignment to $f_1, \ldots, f_k$ and $\delta_1, \ldots, \delta_k$ *is in phase* i, iff $f_i(x) > 0$ and $f_j(x) \leq 0$ for all $j < i$. The condition (4) states that there is always some i such that the multiphase ranking function is in phase i. (5) and (6) state that if we are in a phase $\geq i$, then f_i has to be decreasing by at least $\delta_i > 0$. Note that the 1-phase ranking template coincides with the Podelski-Rybalchenko ranking template.

Multiphase ranking functions are related to *eventually negative expressions* introduced by Bradley, Manna, and Sipma [14]. However, in contrast to our approach, they require a template tree that specifies in detail how each loop transition interacts with each phase.

Lemma 4. *The k-phase ranking template is a linear ranking template.*

Proof. The k-phase ranking template conforms to the syntactic requirements to be a linear ranking template. Let an assignment to the parameters D and the affine-linear function symbols F of the k-phase template be given. Consider the following ranking function with codomain $\omega \cdot k$.

$$\rho(x) = \begin{cases} \omega \cdot (k - i) + \widehat{f}_i(x) & \text{if } f_j(x) \leq 0 \text{ for all } j < i \text{ and } f_i(x) > 0, \\ 0 & \text{otherwise.} \end{cases} \tag{7}$$

Let $(x, x') \in \mathrm{T}$. By Lemma 2, we need to show that $\rho(x') < \rho(x)$. From (4) follows that $\rho(x) > 0$. Moreover, there is an i such that $f_i(x) > 0$ and $f_j(x) \leq 0$ for all $j < i$. By (5) and (6), $f_j(x') \leq 0$ for all $j < i$ because $f_j(x') < f_j(x) - \delta_j \leq 0 - \delta_j \leq 0$, since $f_\ell(x) \leq 0$ for all $\ell < j$.

If $f_i(x') \leq 0$, then $\rho(x') \leq \omega \cdot (k - i) < \omega \cdot (k - i) + \widehat{f}_i(x) = \rho(x)$. Otherwise, $f_i(x') > 0$ and from (6) follows $f_i(x') < f_i(x) - \delta_i$. By Lemma 3, $\widehat{f}_i(x) > \widehat{f}_i(x')$ for the ordinal ranking equivalent of f_i with step size δ_i. Hence

$$\rho(x') = \omega \cdot (k - i) + \widehat{f}_i(x') < \omega \cdot (k - i) + \widehat{f}_i(x) = \rho(x). \qquad \square$$

Example 6. Consider the program from Figure 1. The assignment

$$f_1(q, y) = 1 - y, \qquad f_2(q, y) = q + 1, \qquad \delta_1 = \delta_2 = \tfrac{1}{2}$$

yields a 2-phase ranking function for this program.

Example 7. There are terminating conjunctive linear loop programs that do not have a multiphase ranking function:

$$q > 0 \ \wedge \ q' = q + z - 1 \ \wedge \ z' = -z$$

Here, the sign of z is alternated in each iteration. The function $\rho(q, y, z) = \lceil q \rceil$ is decreasing in every second iteration, but not decreasing in each iteration.

Example 8. Consider the following linear loop program.

$$\begin{aligned} &(q > 0 \ \wedge \ y > 0 \ \wedge \ y' = 0) \\ \vee\ &(q > 0 \ \wedge \ y \leq 0 \ \wedge \ y' = y - 1 \ \wedge \ q' = q - 1) \end{aligned}$$

For a given input, we cannot give an upper bound on the execution time: starting with $y > 0$, after the first loop execution, y is set to 0 and q is set to *some arbitrary value*, as no restriction to q' applies in the first disjunct. In particular, this value does not depend on the input. The remainder of the loop execution then takes $\lceil q \rceil$ iterations to terminate.

However we can prove the program's termination with the 2-phase ranking function constructed from $f_1(q, y) = y$ and $f_2(q, y) = q$.

4.2 Piecewise Ranking Template

The piecewise ranking template formalizes a ranking function that is defined piecewise using affine-linear predicates to discriminate the pieces.

Definition 7 (Piecewise Ranking Template). *We define the* k-piece ranking template *with parameters* $D = \{\delta\}$ *and affine-linear function symbols* $F = \{f_1, \ldots, f_k, g_1, \ldots, g_k\}$ *as follows.*

$$\delta > 0 \tag{8}$$

$$\wedge \bigwedge_{i=1}^{k} \bigwedge_{j=1}^{k} \Big(g_i(x) < 0 \ \vee \ g_j(x') < 0 \ \vee \ f_j(x') < f_i(x) - \delta \Big) \tag{9}$$

$$\wedge \bigwedge_{i=1}^{k} f_i(x) > 0 \tag{10}$$

$$\wedge \bigvee_{i=1}^{k} g_i(x) \geq 0 \tag{11}$$

We call the affine-linear function symbols $\{g_i \mid 1 \leq i \leq k\}$ *discriminators* and the affine-linear function symbols $\{f_i \mid 1 \leq i \leq k\}$ *ranking pieces.*

The disjunction (11) states that the discriminators cover all states; in other words, the piecewise defined ranking function is a total function. Given the k different pieces $f_1, \ldots, f_k$ and a state x, we use f_i as a ranking function only

if $g_i(x) \geq 0$ holds. This choice need not be unambiguous; the discriminators may overlap. If they do, we can use any one of their ranking pieces. According to (10), all ranking pieces are positive-valued and by (9) piece transitions are well-defined: the rank of the new state is always less than the rank any of the ranking pieces assigned to the old state.

Lemma 5. *The k-piece ranking template is a linear ranking template.*

Proof. The k-piece ranking template conforms to the syntactic requirements to be a linear ranking template. Let an assignment to the parameter δ and the affine-linear function symbols F of the k-piece template be given. Consider the following ranking function with codomain ω.

$$\rho(x) = \max \left\{ \widehat{f}_i(x) \mid g_i(x) \geq 0 \right\} \tag{12}$$

The function ρ is well-defined, because according to (11), the set $\{\widehat{f}_i(x) \mid g_i(x) \geq 0\}$ is not empty. Let $(x, x') \in \mathrm{T}$ and let i and j be indices such that $\rho(x) = \widehat{f}_i(x)$ and $\rho(x') = \widehat{f}_j(x')$. By definition of ρ, we have that $g_i(x) \geq 0$ and $g_j(x) \geq 0$, and (9) thus implies $f_j(x') < f_i(x) - \delta$. According to Lemma 3 and (10), this entails $\widehat{f}_j(x') < \widehat{f}_i(x)$ and therefore $\rho(x') < \rho(x)$. Lemma 2 now implies that T is well-founded. □

Example 9. Consider the following linear loop program.

$$\begin{aligned} &(q > 0 \ \wedge \ p > 0 \ \wedge \ q < p \ \wedge \ q' = q - 1) \\ \vee \ &(q > 0 \ \wedge \ p > 0 \ \wedge \ p < q \ \wedge \ p' = p - 1) \end{aligned}$$

In every loop iteration, the minimum of p and q is decreased by 1 until it becomes negative. Thus, this program is ranked by the 2-piece ranking function constructed from $f_1(p, q) = p$ and $f_2(p, q) = q$ with step size $\delta = \frac{1}{2}$ and discriminators $g_1(p, q) = q - p$ and $g_2(p, q) = p - q$. Moreover, this program does not have a multiphase or lexicographic ranking function: both p and q may increase without bound during program execution due to non-determinism and the number of switches between p and q being the minimum value is also unbounded.

4.3 Lexicographic Ranking Template

Lexicographic ranking functions consist of lexicographically ordered components of affine-linear functions. A state is mapped to a tuple of values such that the loop transition leads to a decrease with respect to the lexicographic ordering for this tuple. Therefore no function may increase unless a function of a lower index decreases. Additionally, at every step, there must be at least one function that decreases.

Several different definitions for lexicographic ranking functions have been utilized [12,25,13]; a comparison can be found in [25]. Each of these definitions for lexicographic linear ranking functions can be formalized using linear ranking templates; in this publication we are following the definition of [12].

Definition 8 (Lexicographic Ranking Template). *We define the* k-lexicographic ranking template *with parameters* $D = \{\delta_1, \ldots, \delta_k\}$ *and affine-linear function symbols* $F = \{f_1, \ldots, f_k\}$ *as follows.*

$$\bigwedge_{i=1}^{k} \delta_i > 0 \tag{13}$$

$$\wedge \bigwedge_{i=1}^{k} f_i(x) > 0 \tag{14}$$

$$\wedge \bigwedge_{i=1}^{k-1} \Big(f_i(x') \leq f_i(x) \ \vee \bigvee_{j=1}^{i-1} f_j(x') < f_j(x) - \delta_j \Big) \tag{15}$$

$$\wedge \bigvee_{i=1}^{k} f_i(x') < f_i(x) - \delta_i \tag{16}$$

The conjunction (14) establishes that all lexicographic components $f_1, \ldots, f_k$ have positive values. In every step, at least one component must decrease according to (16). From (15) follows that all functions corresponding to components of smaller index than the decreasing function may increase.

Lemma 6. *The* k*-lexicographic ranking template is a linear ranking template.*

Proof. The k-lexicographic ranking template conforms to the syntactic requirements to be a linear ranking template. Let an assignment to the parameters D and the affine-linear function symbols F of the k-lexicographic template be given. Consider the following ranking function with codomain ω^k.

$$\rho(x) = \sum_{i=1}^{k} \omega^{k-i} \cdot \widehat{f}_i(x) \tag{17}$$

Let $(x, x') \in \mathrm{T}$. From (14) follows $f_j(x) > 0$ for all j, so $\rho(x) > 0$. By (16) and Lemma 3, there is a minimal i such that $\widehat{f}_i(x') < \widehat{f}_i(x)$. According to (15), $\widehat{f}_1(x') \leq \widehat{f}_1(x)$ and hence inductively $\widehat{f}_j(x') \leq \widehat{f}_j(x)$ for all $j < i$, since i was minimal.

$$\begin{aligned} \rho(x') &= \sum_{j=1}^{k} \omega^{k-j} \cdot \widehat{f}_j(x') \leq \sum_{j=1}^{i-1} \omega^{k-j} \cdot \widehat{f}_j(x) + \sum_{j=i}^{k} \omega^{k-j} \cdot \widehat{f}_j(x') \\ &< \sum_{j=1}^{i-1} \omega^{k-j} \cdot \widehat{f}_j(x) + \omega^{k-i} \cdot \widehat{f}_i(x) \leq \rho(x) \end{aligned}$$

Therefore Lemma 2 implies that T is well-founded. $\square$

$$
\begin{aligned}
&\bigwedge_{i=1}^{k}\bigwedge_{j=1}^{\ell}\delta_{i,j} > 0\\
\wedge\ &\bigwedge_{i=1}^{k}\bigvee_{j=1}^{\ell} f_{i,j}(x) > 0\\
\wedge\ &\bigwedge_{i=1}^{k-1}\Big(\big(f_{i,1}(x') \le f_{i,1}(x)\ \wedge\ \bigwedge_{j=2}^{\ell}\big(f_{i,j}(x') \le f_{i,j}(x)\ \vee\ f_{i,j-1}(x) > 0\big)\big)\\
&\quad \vee\ \bigvee_{t=1}^{i-1}\Big(f_{t,1}(x') < f_{t,1}(x) - \delta_{t,1}\ \wedge\ \bigwedge_{j=2}^{\ell}\big(f_{t,j}(x') < f_{t,j}(x) - \delta_{t,j}\ \vee\ f_{t,j-1}(x) > 0\big)\Big)\Big)\\
\wedge\ &\bigvee_{i=1}^{k}\Big(f_{i,1}(x') < f_{i,1}(x) - \delta_{i,1}\ \wedge\ \bigwedge_{j=2}^{\ell}\big(f_{i,j}(x') < f_{i,j}(x) - \delta_{i,j}\ \vee\ f_{i,j-1}(x) > 0\big)\Big)
\end{aligned}
$$

Fig. 2. A k-lexicographic ranking template with ℓ phases in each lexicographic component with the parameters$D = \{\delta_{i,j}\}$ and affine-linear function symbols $F = \{f_{i,j}\}$.

4.4 Composition of Templates

The multiphase ranking template, the piecewise ranking template, and the lexicographic ranking template defined in the previous sections can be used as a 'construction kit' for more general linear ranking templates. Each of our templates contains lower bounds ((4), (10), (11), and (14)) and decreasing behavior ((5), (6), (9), (15), and (16)). We can compose templates by replacing the lower bound conditions and decreasing behavior conditions to affine-linear function symbols in our linear ranking templates with the corresponding conditions of another template. This is possible because linear ranking templates allow arbitrary boolean combination of inequalities and are closed under this kind of substitution. For example, we can construct a template for a lexicographic ranking function whose lexicographic components are multiphase functions instead of affine-linear functions (see Figure 2). This encompasses the approach applied by Bradley et al. [14].

5 Synthesizing Ranking Functions

Our method for ranking function synthesis can be stated as follows. We have a finite pool of linear ranking templates. This pool will include the multiphase, piecewise, and lexicographic ranking templates in various sizes and possibly combinations thereof. Given a linear loop program whose termination we want to prove, we select a linear ranking template from the pool. With this template we build the constraint (2) to the linear ranking template's parameters. If this constraint is satisfiable, this gives rise to a ranking function according to Lemma 2. Otherwise, we try again using the next linear ranking template from the pool

until the pool has been exhausted. In this case, the given linear loop program does not have a ranking function of the form specified by any of the pool's linear ranking templates and the proof of the program's termination failed. See Figure 3 for a specification of our method in pseudocode.

Following related approaches [12,13,14,15,16,17,18,19], we transform the $\exists\forall$-constraint (2) into an $\exists$-constraint. This transformation makes the constraint more easily solvable because it reduces the number of non-linear operations in the constraint: every application of an affine-linear function symbol f corresponds to a non-linear term $s_f^T x + t_f$.

5.1 Constraint Transformation using Motzkin's Theorem

Fix a linear loop program LOOP and a linear ranking template T with parameters D and affine-linear function symbols F. We write LOOP in disjunctive normal form and T in conjunctive normal form:

$$\textsc{loop}(x, x') \equiv \bigvee_{i \in I} A_i \binom{x}{x'} \leq b_i$$

$$\textsc{t}(x, x') \equiv \bigwedge_{j \in J} \bigvee_{\ell \in L_j} \textsc{t}_{j,\ell}(x, x')$$

We prove the termination of LOOP by solving the constraint (2). This constraint is implicitly existentially quantified over the parameters D and the parameters corresponding to the affine-linear function symbols F.

$$\forall x, x'.\ \Big(\big(\bigvee_{i \in I} A_i \binom{x}{x'} \leq b_i \big) \rightarrow \big(\bigwedge_{j \in J} \bigvee_{\ell \in L_j} \textsc{t}_{j,\ell}(x, x') \big) \Big) \tag{18}$$

First, we transform the constraint (18) into an equivalent constraint of the form required by Motzkin's Theorem.

$$\bigwedge_{i \in I} \bigwedge_{j \in J} \forall x, x'.\ \neg \Big(A_i \binom{x}{x'} \leq b_i \ \wedge\ \big(\bigwedge_{\ell \in L_j} \neg \textsc{t}_{j,\ell}(x, x') \big) \Big) \tag{19}$$

Now, Motzkin's Transposition Theorem will transform the constraint (19) into an equivalent existentially quantified constraint.

This $\exists$-constraint is then checked for satisfiability. If an assignment is found, it gives rise to a ranking function. Conversely, if no assignment exists, then there cannot be an instantiation of the linear ranking template and thus no ranking function of the kind formalized by the linear ranking template. In this sense our method is sound and complete.

Theorem 2 (Soundness). *If the transformed $\exists$-constraint is satisfiable, then the linear loop program terminates.*

Theorem 3 (Completeness). *If the $\exists\forall$-constraint* (2) *is satisfiable, then so is the transformed $\exists$-constraint.*

Input: linear loop program LOOP and a list of linear ranking templates $\mathcal{T}$
Output: a ranking function for LOOP or `null` if none is found

```
foreach T ∈ 𝒯 do:
    let φ = ∀x,x'. (LOOP(x,x') → T(x,x'))
    let ψ = transformWithMotzkin(φ)
    if SMTsolver.checkSAT(ψ):
        let (D, F) = T.getParameters()
        let ν = getAssignment(ψ, D, F)
        return T.extractRankingFunction(ν)
return null
```

Fig. 3. Our ranking function synthesis algorithm described in pseudocode. The function `transformWithMotzkin` transforms the $\exists\forall$-constraint φ into an $\exists$-constraint ψ as described in Subsection 5.1.

6 Related Work

The first complete method of ranking function synthesis for linear loop programs through constraint solving was due Podelski and Rybalchenko [17]. Their approach considers termination arguments in form of affine-linear ranking functions and requires only linear constraint solving. We explained the relation to their method in Example 2.

Bradley, Manna, and Sipma propose a related approach for linear lasso programs [13]. They introduce affine-linear inductive supporting invariants to handle the stem. Their termination argument is a lexicographic ranking function with each component corresponding to one loop disjunct. This not only requires non-linear constraint solving, but also an ordering on the loop disjuncts. The authors extend this approach in [14] by the use of *template trees*. These trees allow each lexicographic component to have a ranking function that decreases not necessarily in every step, but *eventually*.

In [16] the method of Podelski and Rybalchenko is extended. Utilizing supporting invariants analogously to Bradley et al., affine-linear ranking functions are synthesized. Due to the restriction to non-decreasing invariants, the generated constraints are linear.

A collection of example-based explanations of constraint-based verification techniques can be found in [18]. This includes the generation of ranking functions, interpolants, invariants, resource bounds and recurrence sets.

In [25] Ben-Amram and Genaim discuss the synthesis of affine-linear and lexicographic ranking functions for linear loop programs over the integers. They prove that this problem is generally co-NP-complete and show that several special cases admit a polynomial time complexity.

References

1. Cook, B., Podelski, A., Rybalchenko, A.: Terminator: Beyond safety. In: CAV. (2006) 415–418

2. Harris, W.R., Lal, A., Nori, A.V., Rajamani, S.K.: Alternation for termination. In: SAS. (2010) 304–319
3. Kroening, D., Sharygina, N., Tonetta, S., Tsitovich, A., Wintersteiger, C.M.: Loop summarization using abstract transformers. In: ATVA. (2008) 111–125
4. Kroening, D., Sharygina, N., Tsitovich, A., Wintersteiger, C.M.: Termination analysis with compositional transition invariants. In: CAV. (2010) 89–103
5. Podelski, A., Rybalchenko, A.: Transition invariants. In: LICS. (2004) 32–41
6. Podelski, A., Rybalchenko, A.: Transition predicate abstraction and fair termination. In: POPL. (2005) 132–144
7. Gupta, A., Henzinger, T.A., Majumdar, R., Rybalchenko, A., Xu, R.G.: Proving non-termination. In: POPL. (2008) 147–158
8. Cook, B., Fisher, J., Krepska, E., Piterman, N.: Proving stabilization of biological systems. In: VMCAI. (2011) 134–149
9. Podelski, A., Wagner, S.: A sound and complete proof rule for region stability of hybrid systems. In: HSCC. (2007) 750–753
10. Albert, E., Arenas, P., Genaim, S., Puebla, G.: Closed-form upper bounds in static cost analysis. J. Autom. Reasoning **46**(2) (2011) 161–203
11. Gulwani, S., Zuleger, F.: The reachability-bound problem. In: PLDI. (2010) 292–304
12. Alias, C., Darte, A., Feautrier, P., Gonnord, L.: Multi-dimensional rankings, program termination, and complexity bounds of flowchart programs. In: SAS. Volume 6337., Springer (2010) 117–133
13. Bradley, A.R., Manna, Z., Sipma, H.B.: Linear ranking with reachability. In: CAV, Springer (2005) 491–504
14. Bradley, A.R., Manna, Z., Sipma, H.B.: The polyranking principle. In: ICALP, Springer (2005) 1349–1361
15. Colón, M.A., Sankaranarayanan, S., Sipma, H.B.: Linear invariant generation using non-linear constraint solving. In: CAV, Springer (2003) 420–432
16. Heizmann, M., Hoenicke, J., Leike, J., Podelski, A.: Linear ranking for linear lasso programs. In: ATVA. (2013)
17. Podelski, A., Rybalchenko, A.: A complete method for the synthesis of linear ranking functions. In: VMCAI, Springer (2004) 239–251
18. Rybalchenko, A.: Constraint solving for program verification theory and practice by example. In: CAV, Springer (2010) 57–71
19. Sankaranarayanan, S., Sipma, H.B., Manna, Z.: Constraint-based linear-relations analysis. In: SAS, Springer (2004) 53–68
20. Grigor'ev, D.Y., N. N. Vorobjov, J.: Solving systems of polynomial inequalities in subexponential time. Journal of Symbolic Computation **5**(1-2) (1988) 37–64
21. Jovanović, D., Moura, L.D.: Solving non-linear arithmetic. In: IJCAR, Springer (2012) 339–354
22. Jech, T.: Set Theory. 3rd edn. Springer (2006)
23. Schrijver, A.: Theory of linear and integer programming. Wiley-Interscience series in discrete mathematics and optimization. Wiley (1999)
24. Ben-Amram, A.M.: Size-change termination, monotonicity constraints and ranking functions. In: CAV, Springer (2009) 109–123
25. Ben-Amram, A.M., Genaim, S.: Ranking functions for linear-constraint loops. In: POPL. (2013)
26. Leike, J.: Ranking function synthesis for linear lasso programs. Master's thesis, University of Freiburg, Germany (2013)