# *Proving Program Termination by DISCOVERER and Complete Discrimination System* *

Yi Li
Automated Reasoning and Cognition Center
CIGIT,CAS
Chongqing, China
liyi@cigit.ac.cn

Yong Feng
Automated Reasoning and Cognition Center
CIGIT,CAS
Chongqing, China
yongfeng@cigit.ac.cn

***Abstract*—Combined with the tools DISCOVERER and CDS, a method is presented to synthesize ranking functions of loop programs. It is shown that the problems of finding ranking functions can be converted to a simpler problem, which can be solved by the complete discrimination system of polynomials (CDS), when DISCOVERER generates only one condition.**

***Keywords- semi-algebraic systems, DISCOVERER , complete discrimination system (CDS), program verification, termination, invariant generation***

## I. INTRODUCTION

Proving total program correctness consists of proving partial correctness, that is, generating loop invariants that is a relation between inputs and outputs, and proving loop termination. Loop invariant generation together with loop termination analysis of programs plays a central role in program verification.

In recent years, with the development of computer algebra, some techniques based symbolic computation have been applied successfully to invariant generation and ranking function discovering. For instance, several important method, such as abstract interpretation [12, 9], quantifier elimination [10, 4, 5] and polynomial algebra [13, 14, 15], have been proposed to generate loop invariants. To guarantee termination, the abstract interpretation methods introduces imprecision by use of an extrapolation operator called widening/narrowing. So this operator often causes the technique to produce weak invariants. [14] presented a method to generate loop invariants represented as polynomial equations. It is first shown that the set of polynomials serving as loop invariants has the algebraic structure of an ideal. Then, Gr$\ddot{o}$bner basis can be used to find invariants. By using an extended Gr$\ddot{o}$bner Basis computation, [15] presented a similar approach to finding polynomial equation invariants in a predefined form. Based on Farkas' Lemma, [7] suggested a technique to find linear inequalities as invariants for linea programs. In 2005, the techniques of parametric abstraction, Lagrangian relaxation and semidefinite programming to generate invariants and ranking functions of polynomial programs were exploited in [9]. This method is very efficient since it does not directly use first-order quantifier elimination. But, [9]'s method is incomplete. That is to say, although there exist ranking functions and invariants of the predefined form, this method may fail to find them. To deal with the problem, [3] proposed an practical and complete approach to generate loop invariants. Their method first reduced the polynomial invariant generation problem to solving semi-algebraic systems. Then the tool DISCOVERER can be employed to solve the resulting semi-algebraic system.

Termination analysis of program loops is very important for ensuring the correct behavior of embedded systems and safety critical software. The traditional method for proving loop termination is by proving that some function of the program variables is well-founded within the loop, which is called ranking function. Various methods have been proposed to attack ranking function discovering problem. Col$o'$n and Sipma first addressed the synthesis of linear ranking functions in deductive verification [6]. Fundamental to their approach is the representation of systems of linear inequalities and sets of linear expressions as polyhedral cones. Then, by the computation of polars intersections and projections of polyhedral cones, a linear ranking function can be found. In [11], Podelski and Rybalchenko provided an efficient and complete synthesis method based linear programming for an unnested program loop. They reduced the synthesis of linear ranking functions to system solving of linear inequalities derived from the program loop. In addition, Tiwari in [16] proved that the termination of a class of single-path loops with linear guards and assignments is decidable. [1] generalized the work of Tiwari, where similar results were derived for termination over the reals. More importantly, it is shown that termination of deterministic linear loops is decidable over the integers in the homogenous case and over the rationals in the general case. By means of the tool DISCOVERER, [2] reduced non-linear ranking function discovering to semi-algebraic system solving. So, their method is complete. However, some universally quantified program variables arise in the resulting condition on parameters. Thus, QEPCAD has to be used to eliminate the remaining quantifiers.

* This work is partially supported by NNSFC-61103110, 90718041 and Shanghai Key Lab. Of Trustworthy Computing.

978-0-7695-4951-4/13 $26.00 © 2013 IEEE
DOI 10.1109/ISISE.2012.15

In this paper, we use semi-algebraic transition system (SATS) [2, 3, 15] that is an extension of algebraic transition system, to represent polynomial programs. For a given polynomial program, a parameterized formula as a possible ranking function is given firstly. As a real algebraic tool, DISCOVERER is a efficient tool to determine whether a given semi-algebraic set (SAS) is feasible or not. However, the SASs DISCOVERER deals with must contain zero-dimensional system of equations. To handle this problem, [2] and [3] first eliminate all the primed variables $V'$ by DISCOVERER to get some conditions. But, some universally quantified program variables remain in the resulting conditions. Then, to get a condition only on the parameters, QEPCAD has to be used to eliminate the remaining quantifiers in their methods. In general, it is difficult to eliminate these universal quantifiers by classic quantifier elimination technique, because of its double-exponential computational complexity in worst case. In the paper, we propose a technique to solve the problem. We first introduce several stack variables $\varepsilon_i s$, which are added to some of constraints to turn inequalities into equations, such that the new SAS contains a zero-dimensional system of equations. Then, solve this new SAS by DISCOVERER and some conditions on parameters and $\varepsilon_i s$ can be derived. The remaining thing is to eliminate $\forall \varepsilon_i \{>,\geq\} 0$ from the conditions. Therefore, the problem of ranking function discovering has been reduced to the problem of elimination of $\forall \varepsilon \{>,\geq\} 0$ from a given inequality $Q(\varepsilon)\{>,\geq\} 0$. Clearly, the complete discrimination system (CDS) of polynomial is a powerful tool for solving this kind of problem. And some elegant results have been presented in [18, 20, 22]. Our method is also complete due to the features of DISCOVERER and CDS.

The rest of the paper is organized as follows: Section 2 presents a brief review of the theories of CDS. In Section 3, we illustrate this method on several examples. Finally, Section 4 draws a summary and discusses future work.

## II. COMPLETE DISCRIMINATION SYSTEM OF POLYNOMIAL

In this section, we describe some important theories on complete discrimination system (CDS)[18,20] that is the foundation stone of our technique. In general, a CDS is a set of explicit expressions in terms of the coefficients that are sufficient for determining the numbers and multiplicities of the real and imaginary roots for polynomials with symbolic coefficients. We first introduce some basic notions.

### *A. Main Results on Complete Discrimination System*

**Definition 1 [Discrimination Matrix]** Given a Polynomial with general symbolic coefficients,

$$f(x) = a_0 x^n + a_1 x^{n-1} + \cdots + a_n,$$

the following $(2n+1)\times(2n+1)$ matrix in terms of the coefficients,

$$\begin{bmatrix}
a_0 & a_1 & a_2 & \cdots & a_n & & & \\
0 & na_0 & (n-1)a_1 & \cdots & a_{n-1} & & & \\
 & a_0 & a_1 & \cdots & a_{n-1} & a_n & & \\
 & 0 & na_0 & \cdots & 2a_{n-2} & a_{n-1} & & \\
 & & & \cdots & \cdots & & & \\
 & & & \cdots & \cdots & & & \\
 & & & & a_0 & a_1 & \cdots & a_n \\
 & & & & 0 & na_0 & \cdots & a_{n-1} \\
 & & & & & a_0 & a_1 & \cdots & a_n
\end{bmatrix}$$

is called the discrimination matrix of $f(x)$, and denoted by Discr(f). Let $\{d_1, d_2, \ldots, d_{2n+1}\}$ be the principal minor sequence of Discr(f). Where $d_i$ denotes the determinant of the submatrix of Discr(f).

**Definition 2 [Discriminant Sequence]** Let $D_k = d_{2k}$ for $k = 1, \ldots, n$. We called the n-tuple $\{D_1, D_2, \ldots, D_n\}$ the discriminant sequence of polynomial $f(x)$.

**Definition 3 [Sign List]** With the notations as above, we call the list $[sign(D_1), sign(D_2), \ldots, sign(D_n)]$, the sign list of a given sequence $\{D_1, D_2, \ldots, D_n\}$

**Definition 4 [Revised Sign List]** Given a sign list $[s_1, s_2, \ldots, s_n]$, the revised sign list $[\varepsilon_1, \varepsilon_2, \ldots, \varepsilon_n]$ is a new list constructed by replacing some of $0's$ in the former list with 1 or -1 according to the following rules:

- If $[s_i, s_{i+1}, \ldots, s_{i+j}]$ is a section of the given list, where $s_i \neq 0; s_{i+1} = s_{i+2} = \cdots = s_{i+j-1} = 0;$ then, we replace the subsection $[s_{i+1}, s_{i+2,\cdots,}]$ with $[-s_i, -s_i, s_i, s_i, -s_i, \ldots]$ keeping the number of terms. Thus, the revised section does no longer contain $0's$.
- Otherwise, let $\varepsilon_k = s_k$, i.e. no changes for other terms.

For example, given a sign list $[1,0,0,0,-1,0,1]$, then, the corresponding revision version is $[1,-1,-1,1,-1,1,1]$ according to Definition 4. By the following theorem, we can determine the numbers and multiplicities of the real and imaginary roots of a given polynomial.

**Theorem 5** With the above notion. If the number of the sign changes of the revised sign list of $\{D_1, D_2, \cdots, D_n\}$ is $v$, then the number of the pairs of distinct conjugate imaginary roots of $f(x)$ equals $v$. Furthermore, if the number of non-vanishing members of the revised version is $l$, then then number of the distinct real roots of $f(x)$ equals $l-2v$.

For saving space, we omit all proofs of the results stated above. For more details, please refer to [18].

**Example 1.** Given a polynomial,

$$f(x) = x^{16} - 3x^{10} + 78x^9 - x^6 + 5x^5 + 31x^3 - x + 1.$$

First, the sign list of the discriminant sequence $f(x)$ is derived , as follows:

$$[1,0,0,0,0,1,-1,-1,1,1,-1,-1,1,1,1,-1]$$

Then, by Definition 4, the corresponding revision version is:

$$[1,-1,-1,1,1,1,-1,-1,1,1,-1,-1,1,1,1,-1]$$

Thus, by Theorem 5, we know that $f(x)$ has 7 pairs of distinct conjugate imaginary roots and 2 distinct real roots.

## III. PROGRAM VERIFICATION VIA DISCOVERER AND CDS

In this section, some necessary notations are stated firstly. Then, we illustrate the technique of synthesizing ranking functions by two examples.

**Definition 6** A semi-algebraic transition system (SATS) is a quintuple $<V,L,T,l_0,\Theta>$, where $V$ is a set of program variables, $L$ is a set of locations, and $T$ is a set of transitions. Each transition $\tau \in T$ is a quadruple $<l_1,l_2,\rho_\tau,\theta_\tau>$, where and $l_2$ are the pre- and post-locations of the transition, $\rho_\tau \in CPF(V,V')$ is the transition relation, and $\theta_\tau \in CPF(V)$ is the guard of the transition. Where $V'$ denotes the next-state variables. $l_0$ represents the initial location, and $\Theta \in CPF(V)$ is the initial condition.

**Definition7 [Ranking Function]** Assume $P=<V,L,T,l_0,\Theta>$ is an SATS (semi- algebraic transition system). A ranking function is a function

$\gamma : Val(V) \to R^+$

such that the following conditions are satisfied:

**Initial Condition**: $\Theta(V_0) \quad \gamma(V_0) \geq 0.$

**Decreasing Condition**: There exists $C \in R^+$ such that $M > 0$ and for any transition circle at $l_0$

$$l_0 \xrightarrow{\tau_1} l_1 \xrightarrow{\tau_2} \cdots \xrightarrow{\tau_{n-1}} l_{n-1} \xrightarrow{\tau_n} l_0,$$
$$\rho_{\tau_1;\tau_2;\cdots\tau_n}(V,V') \wedge \theta_{\tau_1;\tau_2;\cdots\tau_n}(V),$$
$$\gamma(V) - \gamma(V') \geq M \wedge \gamma(V') \geq 0.$$

### A. Synthesis of Non-linear Ranking Function

Several approaches of synthesizing linear ranking function have been proposed in [6, 8]. But in many cases, a program may has only non-linear ranking function. In this subsection, by means of DISCOVERER and CDS, we show how to synthesize non-linear ranking function in the following example.

**Example 2.** Consider a program shown in Figure 1.

```
(x, y) ∈ V_R(x^2 − y)
        while x + y + 3 < 0 do
          x := 2x^2 − y, y := 2y
        endwhile
```

Figure 1. Example of a loop program

```
P = {V = {x, y},
T = {τ}, Θ = {x^2 − y = 0}
τ_1 = {x + y + 3 < 0, x′ − 2x^2 + y = 0,
                      y′ = 2y}
```

Figure 2. Semi-algebraic transition system

To find the ranking function of the loop, we first predefine a template of ranking function, that is, $\gamma = ax + by^2 + c.$ Assume that $ab \neq 0$. Semi-algebraic transition system (SATS) can be represented as in Figure 2.

*Step*1. Encoding initial condition, that is, $\Theta \mapsto \gamma \geq 0$. This is equivalent to that $\Theta \wedge \gamma < 0$ has no real solution. Namely, the below semi-algebraic system has no real solution, $\{x^2 - y = 0 \wedge ax + by^2 + c < 0\}$. But, the tool DISCOVERER can not directly deal with this kind of semi-algebraic set since it contains no zero-dimensional system of equations. Whereas, it is easy to prove that the above semi-algebraic system is equivalent to the following semi-algebraic system, which can be dealt with by DISCOVERER well, $\{x^2 - y = 0, ax + by^2 + c + \varepsilon_1 = 0, \varepsilon_1 > 0\}$.

Thus, encoding initial condition is equivalent to finding the conditions on parameters $a,b,c$ such that the new semi-algebraic system has no real solution. Invoking the command in DISCOVERER,

$$tofind([x^2 - y, ax + by^2 + c + \varepsilon_1],[\,],[\varepsilon_1],[\,],[x,y],[a,b,c,\varepsilon_1],0),$$

we obtain a condition on parameters $a,b,c,\varepsilon_1$, as follows.

$$Cond_1 = 27a^4 - 256bc^3 - 768\varepsilon_1 bc^2 - 768bc\varepsilon_1^2 - 256b\varepsilon_1^3 < 0 \wedge b \neq 0. \quad (1)$$

Here, parameters $\varepsilon_1$ must be eliminated from $Cond_1$. As shown in [3], we can use QEPCAD to eliminate the universally quantified parameters $\varepsilon_1$. However, it is not necessary for this kind of problem to use those tools based on quantifier elimination technique, when some universal quantifiers need to be eliminated, since its high computational complexity. In contrast to quantifier elimination technique, we here use CDS to eliminate these universally quantified parameters $\forall \varepsilon_1 > 0$ in this example. Clearly, the left-hand side of $Cond_1$ is regarded as a polynomial in $\varepsilon_1$. Based on CDS, more concise results have been presented in [18,22]. We here state these results below.

**Theorem 8**[22] *Given a polynomial of 3 degrees*

$$Q(\lambda) = \lambda^3 + p\lambda^2 + q\lambda + r.$$

Then, $(\forall \lambda > 0)(Q(\lambda) > 0)$ is equivalent to

$$(p \geq 0 \wedge q \geq 0 \wedge r \geq 0) \vee (D_3 < 0 \wedge r \geq 0).$$

Where, $D_3 = -27r^2 + 18rpq + p^2q^2 - 4p^3r - 4q^3.$

By the above theorem 8, eliminating $\forall \varepsilon_1 > 0$ from $Cond_1$, we get the following conditions

$$b \neq 0 \wedge -27a^4b + 256c^3b^2 \geq 0. \quad (2)$$

*Step*2. Encoding Decreasing Condition.

Firstly, $\rho \wedge \theta \quad \gamma(\{x',y'\}) \geq 0$ if and only if

$$\{x+y+3<0, x'-2x^2+y=0, y'-2y=0, \quad (3)$$
$$ax'+by'^2+c<0\}$$

has no real solution. It is easy to see that the variables $x', y'$ can be replaced with two polynomials in $x, y$. Thus, the above semi-algebraic system can be rewritten in the following form.

$$\{x+y+3<0, a(2x^2-y)+b(2y)^2+c<0\}.$$

Introducing the new variables $\varepsilon_1, \varepsilon_2$, we have

$$\{x+y+3+\varepsilon_1=0, a(2x^2-y)+b(2y)^2+c+\varepsilon_2=0, \quad (4)$$
$$\varepsilon_i>0, i=1,2\}.$$

Calling the command in DISCOVERER,

$$tofind([x+y+3+\varepsilon_1, a(2x^2-y)+b(2y)^2+c+\varepsilon_2],$$
$$[\,],[\varepsilon_1,\varepsilon_2],[\,],[x,y],[a,b,c,\varepsilon_1,\varepsilon_2],0),$$

we get

$$Cond=23a^2+8\varepsilon_1 a^2+192b\varepsilon_1 a+32ab\varepsilon_1^2+8ac+8a\varepsilon_2+288ba+$$
$$16bc+16\varepsilon_2 b>0\wedge a+2b\neq 0.$$

In terms of the arguments mentioned above, we first eliminate $\forall\varepsilon_2>0$ from *Cond*. The left-hand side of *Cond* is a polynomial in $\varepsilon_2$ of degree 1. Thus,

$\forall\varepsilon_2>0(Cond(\varepsilon_2)>0)$ is equivalent to $a+2b>0\wedge$

$$192b\varepsilon_1 a+32b\varepsilon_1^2 a+16cb+288ba+8ca+23a^2+8\varepsilon_1 a^2\geq 0. \quad (5)$$

The remaining thing is to eliminate $\forall\varepsilon_1>0$ from (5). The following results based on CDS are very useful to eliminate $\varepsilon_1$.

**Lemma 9**[17] Let $\{d_1, d_2, ..., d_{2n+1}\}$ be the principal minor sequence of Discr(f), the discrimination matrix of polynomial $f(x)$ with $f(0)\neq 0$. Denote the number of sign changes and the number of non-vanishing members of the revised sign list of sequence $\{d_1d_2, d_2d_3, ..., d_{2n}d_{2n+1}\}$ by $v$ and $l$, respectively. Then, the number of the distinct negative roots of $f(x)$ equals $\frac{l-2v}{2}$.

According to Lemma 9 as before, we give the following theorem.

**Theorem 10** *Given a quadric polynomial of real coefficients,* $Q(\lambda)=\mu_1\lambda^2+\mu_2\lambda+\mu_3$,

with $\mu_1\neq 0$, then $(\forall\lambda>0)Q(\lambda)>0$ is equivalent to

$$\mu_1>0\wedge((\mu_1\mu_2>0\wedge\Delta=0)$$
$$\vee(\mu_1\mu_2\geq 0\wedge\Delta\neq 0\wedge\mu_1\mu_3>0)\vee(\mu_1\mu_3>0\wedge\Delta<0))$$

where $\Delta=\mu_2^2-4\mu_1\mu_3$.

*Proof:* The proof is very simple according to CDS and Lemma 9. One just needs to find some conditions under which $Q(\lambda)$ has no positive roots. Let $Q'(\lambda)=Q(-\lambda)$. Hence,by Lemma 14, we get the following several cases of sign list of $Q'(\lambda)$:

$$\{[1,-1,0,0],[1, sign\leq 0, sign\in\{\geq 0,\leq 0\},1],$$
$$[1, sign\geq 0, sign\leq 0,1]\}.$$

This completes the proof of the theorem. □

By Theorem 10, eliminating $\forall\varepsilon_1>0$ from (5), we have

$$a+2b>0\wedge ab>0\wedge(ab(24ba+a^2)>0\wedge\Delta=0)\vee$$
$$(ab(24ba+a^2)\geq 0\wedge\Delta\neq 0\wedge ab(16cb+288ba+8ca$$
$$+23a^2)>0)\vee(ab(16cb+288ba+8ca+23a^2)>0$$
$$\wedge\Delta<0).$$

Where $\Delta=a(-576ab^2-44a^2b+a^3-64b^2c-32abc)$.

Secondly, $\rho\wedge\theta\ \ \gamma(\{x,y\})-\gamma(\{x',y'\})\geq M>0$ if and only if

$$\{x+y+3<0, x'-2x^2+y=0, y'-2y=0, \quad (6)$$
$$a(x-x')+b(y^2-y'^2)<M, M>0\}.$$

has no real solution. By computation, we have

$$Cond=2\varepsilon_1 a^2+6a^2+39b\varepsilon_1 a-2a\varepsilon_2+2aM+63ba$$
$$+6ab\varepsilon_1^2-3\varepsilon_2 b+3bM>0\wedge 2a+3b\neq 0.$$

By the similar method as above, it follows that

$$2a+3b<0\wedge ab>0\wedge(((2a^2+39ba)>0\wedge\Delta=0) \quad (7)$$
$$\vee(\wedge 2a^2+39ba\geq 0\wedge 6a^2+2aM+63ba+3bM>0$$
$$\wedge\Delta\neq 0)\vee((6a^2+2aM+63ba+3bM)>0\wedge\Delta<0)).$$

Where $\Delta=a(2a+3b)(2a^2+3ba-24bM)$.

Solving (2) ∧ (6) ∧ (7), we find no sample points by CAD technique. Thus, we can draw a conclusion that the program has no ranking function in predefined form.

**Remark.** Our method is different from the approach presented in [2,3]. This difference mainly is embodied in two aspects: First, introducing new variables may construct a new semi-algebraic set with system of polynomial equations, which can be dealt with well by DISCOVERER; Second, the primary problem of getting the conditions only on parameters can be reduced to a simpler problem of eliminating $\forall\varepsilon\{>,\geq\}0$ from $Q(\lambda)\{>,\geq 0\}0$. Thus, CDS can be utilized directly to eliminate the remaining universally quantified quantifiers.Certainly, we also can use the methods proposed in [2,3]. However, if we do like this, we may encounter the troublesome problem of eliminating $\forall var\in U(f(var)\{>,\geq\}0)$ from $Q(var)\{>,\geq\}0$. Where, var denotes a program variable and $f(var)$ is a polynomial in $var$. Clearly, it is difficult to solve this kind of problem. So, in [2,3], the authors have to use the tool QEPCAD to eliminate the remaining program variables when this case happens.

## IV. Conclusion and Future Work

DISCOVERER is a powerful tool for determining whether a given semi-algebraic set is feasible or not. More importantly, it can determine the conditions on parameters such that the number of the distinct real solutions of a semi-algebraic set equals to an integer.

But, This tool requires that the semi-algebraic set it deals with must contain a zero-dimensional system of equations. Based on this point, in this paper, introducing stack variables, we get rid of this restriction such that DISCOVERER can deal with more wide semi-algebriac

problems. Besides, CDS is also utilized to eliminate universally quantified stack variables. This leads to lower computational complexity since CDS is effective tool for determining the numbers of the real roots and imaginary roots of polynomials with parametric coefficients. In this paper, two examples are presented to illustrate our technique. More experimental results show that our technique is effective, especially when DISCOVERER generates only one condition.

When DISCOVERER generates more than one condition, we still can employ CDS completely to further analyse these conditions. Certainly, this requires more skills. And corresponding arguments about this case will be reported in another paper. In addition, our future work is also going to use technique to simplify the final results such that the desired results seem to be more laconic.

## ACKNOWLEDGMENT

We would like to thank Prof. Yang Lu for encouraging us to work on the problem.

## REFERENCES

[1] Braverman, M.: Termination of integer linear programs. In: Ball, T., Jones, R.B. (eds.) CAV 2006. LNCS, vol. 4144, pp. 372-385. Springer, Heidelberg ,2006

[2] Chen, Y., Xia, B., Yang, L., Zhan, N., Zhou, C.: Discovering non-linear ranking functions by solving semi-algebraic systems. In:L ICTAC 2007. LNCS, Springer, Heidelberg ,2007

[3] Chen, Y.H., Xia, B., Yang, L., Zhan, N.J.: Generating Polynomial Invariants with DISCOVERER and QEPCAD. In: Jones, C.B., Liu, Z. Woodcock, J. (eds.) LNCS 4700, pp. 67-82, 2007. Springer, Heidelberg ,2007

[4] Collins, G.E.: Quantifier elimination for real closed fields by cylindrical algebraic decomposition. In: Brakhage, H. (ed.) Automata Theory and Formal Languages. LNCS, vol. 33, pp. 134-183. Springer, Heidelberg ,1975

[5] Collins, G.E., Hong, H.: Partial cylindrical algebraic decomposition for quantifier elimination. J. of Symbolic Computation 12, 299-328 ,1991

[6] Col$O'$n, A.M., Sipma, H.B.: Practical methods for proving program termination. In: Brinksma, D. and Larsen, K.G. (Eds.): CAV 2002, LNCS 2404, pp. 442-454, 2002. Springer, Heidelberg ,2002

[7] Col$O'$n, M., Sankaranarayanan, S., Sipma, H.B.: Linear invariant generation using non-linear constraint solving. In: Hunt Jr., W.A., Somenzi, F. (eds.) CAV 2003. LNCS, vol. 2725, pp. 420-432. Springer, Heidelberg ,2003

[8] Col$O'$n, M., Sipma, H.B.: Synthesis of linear ranking functions. In: Margaria, T, Yi, W. (eds.) ETAPS 2001 and TACAS 2001. LNCS, vol. 2031, pp. 67-81. Springer, Heidelberg ,2001

[9] Cousot, P.: Proving program invariance and termination by parametric absstraction, Langrangian Relaxation and semi-definite programming. In: Cousot, R. (ed.) VMCAI 2005. LNCS, vol. 3385, pp. 1-24. Springer, Heidelberg ,2005

[10] Kapur, D.: Automatically generating loop invariants using quantifier elimination. In: Proc, IMACS Intl. Conf. On Applications of Computer Algebra (ACA'04), Beaumont, Texas (2004)

[11] Podelski, A., Rybalchenko, A.: A complete method for the synthesis of linear ranking functions. In: Steffen, B., Levi, G. (eds.) VMCAI 2004. LNCS, vol. 2937, pp. 239-251. Springer, Heidelberg ,2004

[12] Rodriguez-Carbonell, E., Kapur, D.: An abstract interpretation approach for automatic generation of polynomial invariants. In: Giacobazzi, R. (ed.) SAS 2004. LNCS, vol. 3148, pp. 280-295. Springer, Heidelberg ,2004

[13] Rodriguez-Carbonell, E., Kapur, D.: Generating all polynomial invariants in simple loops. Journal of Symbolic Computation 42, 443-476 ,2007

[14] Rodriguez-Carbonell, E., Kapur, D.: Automatic generation of polynomial loop invariants: algebraic foundations. In: Proc. Intl. Symp on Symbolic and Algebraic Computation (ISSAC'04) ,2004

[15] Sankaranarayanan, S., Sipma, H.B., Manna, Z.: Non-linear loop invariant generation using Groebner basis. In: ACM POPL'04, pp. 318-329 ,2004

[16] Tiwari, A.: Termination of linear programs. In: Alur, R., Peled, D.A. (eds.) CAV 2004. LNCS, vol. 3114, pp. 70-82. Springer, Heidelberg ,2004

[17] Xia, B., Yang, L.: An algorithm for isolating the real solutions of semi-algebraic systems. J. Symbolic Computation 34, 461-477 ,2002

[18] Yang, L.: Recent advances on determining the number of real roots of parametric polynomials. J. Symbolic Computation 28, 225-242 ,1999

[19] Yang, L., Hou, X., Xia, B.: A complete algorithm for automated discovering of a class of inequality-type theorems. Sci, In China (Ser. F) 44, 33-49 ,2001

[20] Yang, L., Hou, X., Zeng, Z.: A complete discrimination system for polynomials. Science in China (Ser. E) 39, 628-646 ,1996

[21] Yang, L., Xia, B.: Real solution classifications of a class of parametric semi-algebraic systems. In: Proc. of Int'l Conf. On Algorithmic Algebra and Logic, PP. 281-289 ,2005

[22] Yang, L., Zhan, N., Xia, B., zhou, C.: Program verification by using DISCOVERER. In: Meyer, B and Woodcock, J. (eds.) Verified Software, LNCS 4171,pp.528-538,2008