基于 SVM 算法的循环程序的终止性证明

硕士生：孙学超
导　师：张立军　研究员

中国科学院软件研究所
中国科学院大学

Verification

```
assume(True);
int x = 1;
while(x>0)
{
    x++;
}
assertion(x<=0);
```

- Step 1: Can the program reach the assertion?
- Step 2: Does the assertion hold?

Verification

```
assume(True);
int x = 1;
while(x>0)
{
  x++;
}
assertion(x<=0);
```

- It is very important to check whether the program is TERMINATING in the verification problem.

Bad news & Good news

- In theory,
  the termination problem of programs has been proven to be undecidable.
- In practice,
  just return "UNKNOWN" when we can not prove whether it is terminating.

Bad news & Good news

- In theory,
  the termination problem of programs has been proven to be undecidable.
- In practice,
  just return "UNKNOWN" when we can not prove whether it is terminating.

Goal:

For a given program, we try to avoid "UNKNOWN" result as much as possible.

Background

- It attracts many researchers to work on the termination of programs.
- Ultimate Automizer, one of the leading tools in program analysis according to the outcomes of the SV-COMP competitions.
- The termination of loops is at the core of the termination analysis techniques used in Ultimate Automizer.

Concern

```
while(g_1) \\Loop 1
{
    ....;
    ....;
    ....;
}
```

the termination of simple loop programs, i.e., no nested loops.

Ranking function (RF)

- Informal definition

$$f : S \to \mathbb{R}$$

where S is the set of program states, $\mathbb{R}$ is a well-founded ordered set.

- Decrease condition

$$\forall x, x' : f(x) - f(x') \geq \delta > 0$$

where $\delta \in \mathbb{R}$, x is current program state, x' is the program state after updating.

- Bounded condition

$$\forall x : f(x) \geq C$$

where $C \in \mathbb{R}$.

Ranking function (RF)

Thm.

A loop program P is terminating, if P has a ranking function f(x).

Synthesis of ranking functions

- Termination of programs is undecidable.
- Synthesis of ranking function is, however, decidable, given certain classes of ranking functions.

Related works

- Linear program & Single phase linear ranking function
  - In 2002, Colón and Sipma synthesized linear ranking functions for linear-constraint loops.
  - In 2004, a complete and efficient solution was proposed by Podelski and Rybalchenko.
  - ...
- Linear program & K phases ranking function
  - In 2005, Bradley et al. showed how to synthesize lexicographic linear ranking functions (LLRFs).
  - In 2014, Ben-Amram and Genaim provided a complete polynomial-time solution for MΦRFs with bounded depth.
  - ...
- Polynomial program & Single phase ranking function
  - In 2005, Cousot made use of parametric abstraction and SDP to compute ranking functions of loops.
  - In 2019, Yuan et al. proposed a ranking function detection method exploiting SVM .
  - ...

Our contributions

- Linear program & Single phase linear ranking function
- Linear program & K phases ranking function
- Polynomial program & Single phase ranking function
- Polynomial program & K phases ranking function
    - Based on SVM to synthesize nested ranking functions.
    - Provide the comprehensive empirical evaluation.

Preliminaries

Def. (Loop Program)

A loop program $\Omega(\mathrm{x}, \mathrm{x}')$ is a binary relation with free variables x and x', where x is the current state, and x' is the next state.

```
while (x>0)
{
   x++;
}
```

$$\Omega(\mathrm{x}, \mathrm{x}') \triangleq \mathrm{x} > 0 \wedge \mathrm{x}' = \mathrm{x} + 1$$

Preliminaries

Def. (k-Nested Ranking Function)

Given a loop program Ω, let $k \in \mathbb{N}_{>0}$ and, for each $i \in \{1, \ldots, k\}$, $f_i(x)$ be a polynomial or an algebraic fraction over the program variables x. We call the k-tuple $\langle f_1, f_2, \ldots, f_k \rangle$ a k-nested ranking function of Ω if the following condition holds for a set of parameters $\{\, C_i \in \mathbb{R}_{>0} \mid 1 \leq i \leq k+1 \,\}$:

$$\forall (x, x') \in \Omega : \begin{cases} f_1(x) - f_1(x') \geq C_1 \\ f_2(x) - f_2(x') + f_1(x) \geq C_2 \\ \vdots \\ f_k(x) - f_k(x') + f_{k-1}(x) \geq C_k \\ f_k(x) \geq C_{k+1} \end{cases}$$

Example

```
int q,y;
while(q>0)
{
  q = q-y;
  y = y+1;
}
```

- Single phase linear ranking function does not work.
- However, it has a 2-nested ranking function.

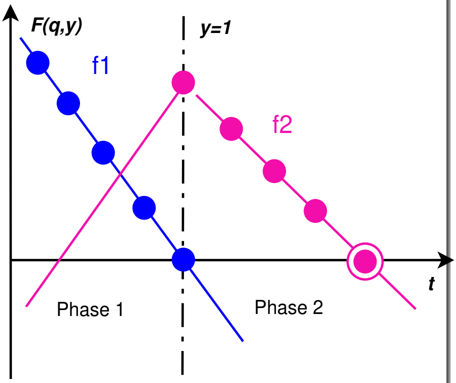
$$f_1(q,y) = 1 - y, f_2(q,y) = q + 1, C_1 = C_2 = C_3 = 1;$$

Example

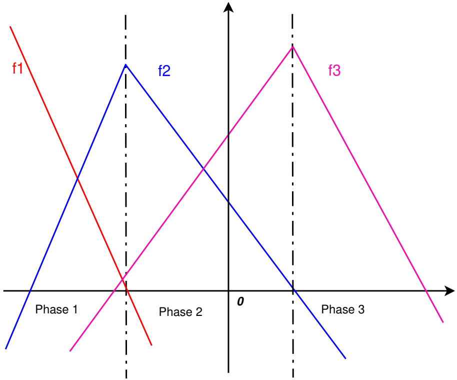
Loop Program

```
int q,y;
while(q>0)
{
   q = q-y;
   y = y+1;
}
```

Phases

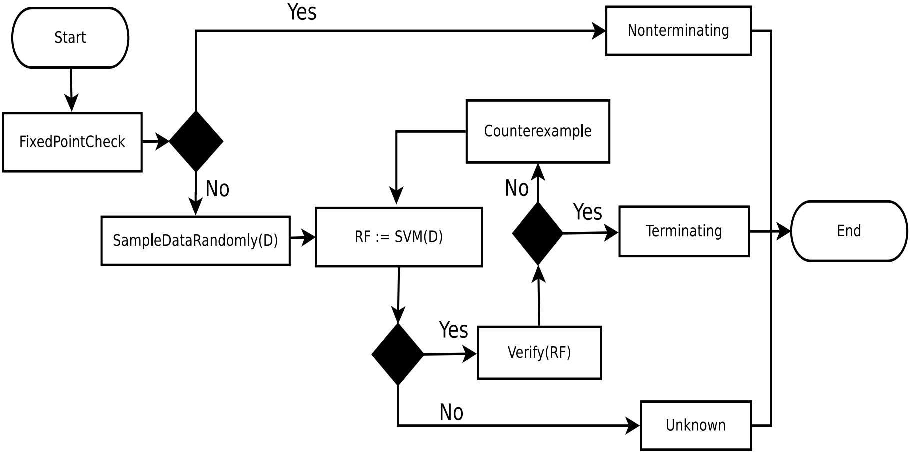


Intuition



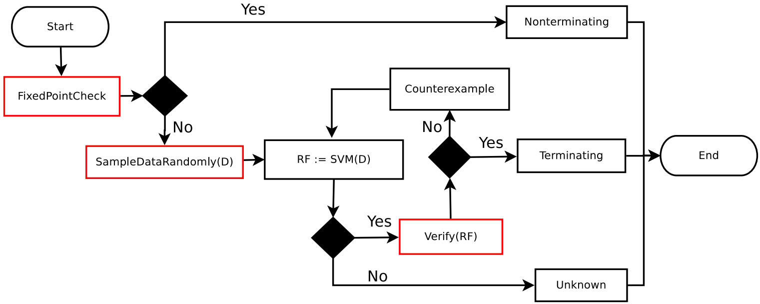
- We allow that the ranking function can increase in the previous phases.
- It will finally decrease in their own phase and the following phases.

Overview of the Algorithm



Problems

- How to check the fixed point?
- How to verify the correctness of learned ranking function?
- How to construct the data set?



How to check the fixed point?

Def. (Fixed point)

Given a loop program specified by Ω, we say that $x \in \mathbb{R}^n$ is a fixed point of the loop if $(x, x) \in \Omega$.

Example

```
while (x>0 && y>0)
{
  x = x+y;
  x = x-y;
}
```

$$X' = (x', y') = ((x+y) - y, y) = (x, y) = X$$

$$\Longrightarrow \forall X \in \mathbb{R}^2_{>0}, (X, X) \in \Omega$$

$$\Longrightarrow \forall X \in \mathbb{R}^2_{>0}, X \text{ is a fixed point.}$$

How to verify the correctness of learned ranking functions?

- Decreased condition & Bounded condition
- SMT tools, like Z3, etc.

Check for single phase ranking function

$$\forall (x, x') \in \Omega : f(x) \geq C \wedge f(x) - f(x') \geq \delta$$

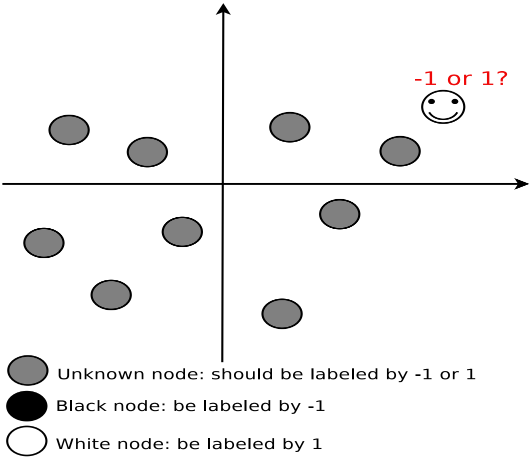
How to verify the correctness of learned ranking functions?

Check for nested ranking functions

$$\forall (x, x') \in \Omega : \begin{cases} f_1(x) - f_1(x') \geq C_1 \\ f_2(x) - f_2(x') + f_1(x) \geq C_2 \\ \vdots \\ f_k(x) - f_k(x') + f_{k-1}(x) \geq C_k \\ f_k(x) \geq C_{k+1} \end{cases}$$

How to construct the data set?

- Relation between the SVM & the sampled data set.
- Relation between the SVM & the ranking function.



Ranking function

Let $\langle f_1, \ldots, f_k \rangle$ be a k-tuple representing a k-nested ranking functions;

$$f_j(x) = a_j^T \cdot U_j(x)$$

where $a_j = (a_{j,1}, \ldots, a_{j,s_j})$ is a real vector of coefficients and $U_j(x) = (U_{j,1}(x), \ldots, U_{j,s_j}(x))^T$ is an s_j-tuple with $U_{j,i}(x) = \frac{q_{j,i}(x)}{p_{j,i}(x)}$, where $q_{j,i}(x), p_{j,i}(x) \in \mathbb{R}[x]$, for $i \in \{1, \ldots, s_j\}$.

Ranking function

Exp.

$$\mathrm{f_j(x)} = 3x^2 - 4xy + \frac{5y^3}{3x^3 + 2y + 1} + 7$$

$$\mathrm{f_j(x)} = \mathrm{a_j^T} \cdot \mathrm{U_j(x)}$$

where $\mathrm{a_j^T} = (3, -4, 5, 7)$ and $\mathrm{U_j(x)} = (x^2, xy, \frac{y^3}{3x^3+2y+1}, 1)^T$

Nested ranking function

$$\forall (x, x') \in \Omega : \begin{cases} f_1(x) - f_1(x') \geq C_1 \\ f_2(x) - f_2(x') + f_1(x) \geq C_2 \\ \vdots \\ f_k(x) - f_k(x') + f_{k-1}(x) \geq C_k \\ f_k(x) \geq C_{k+1} \end{cases}$$

$\Longrightarrow$

$$\forall (x, x') \in \Omega : \begin{cases} a_1^T \cdot (U_1(x) - U_1(x')) \geq C_1 \\ a_2^T \cdot (U_2(x) - U_2(x')) + a_1^T \cdot U_1(x) \geq C_2 \\ \vdots \\ a_k^T \cdot (U_k(x) - U_k(x')) + a_{k-1}^T \cdot U_{k-1}(x) \geq C_k \\ a_k^T \cdot U_k(x) \geq C_{k+1} \end{cases}$$

Substitute

$$
G_1(x,x') \mapsto \begin{pmatrix} 0 \\ \vdots \\ 0 \\ 0 \\ U_1(x) - U_1(x') \end{pmatrix}, \; G_2(x,x') \mapsto \begin{pmatrix} 0 \\ \vdots \\ 0 \\ U_2(x) - U_2(x') \\ U_1(x) \end{pmatrix}, \; \cdots
$$

$$
\cdots, \; G_k(x,x') \mapsto \begin{pmatrix} U_k(x) - U_k(x') \\ U_{k-1}(x) \\ 0 \\ \vdots \\ 0 \end{pmatrix}, \; G_{k+1}(x,x') \mapsto \begin{pmatrix} U_k(x) \\ 0 \\ 0 \\ \vdots \\ 0 \end{pmatrix}
$$

Constraint

$$\forall (x, x') \in \Omega : \begin{cases} (a_k^T, \cdots, a_1^T) \cdot G_1(x, x') \geq C_1 \\ (a_k^T, \cdots, a_1^T) \cdot G_2(x, x') \geq C_2 \\ \vdots \\ (a_k^T, \cdots, a_1^T) \cdot G_k(x, x') \geq C_k \\ (a_k^T, \cdots, a_1^T) \cdot G_{k+1}(x, x') \geq C_{k+1} \end{cases} \tag{1}$$

where $C_i > 0, i \in 1..k+1$.

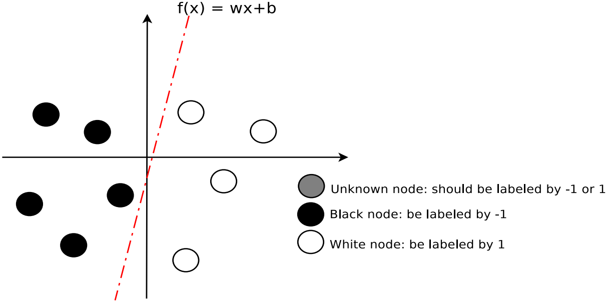
SVM

- Positive examples:

$$f(x) = wx + b > 0$$

- Negative examples:

$$f(x) = wx + b < 0$$



Positive examples

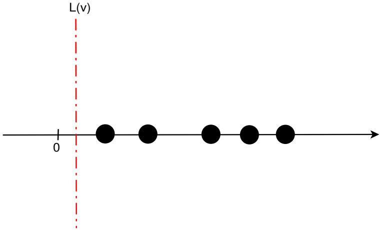
$$\forall(x, x') \in \Omega : \begin{cases} w \cdot G_1(x, x') + b > 0 \\ w \cdot G_2(x, x') + b > 0 \\ \vdots \\ w \cdot G_k(x, x') + b > 0 \\ w \cdot G_{k+1}(x, x') + b > 0 \end{cases}$$

where $w = (a_k^T, \cdots, a_1^T), b = 0$.

Thm.

Given a loop specified by Ω, it have nested polynomial ranking functions as defined in Eq. 1 if and only if there exists a hyperplane $L(v)$ strictly separating the origin $O \in \mathbb{R}^m$ from $G(\Omega) \subseteq \mathbb{R}^m$.

Proof.



□

Example

```
int q,y;
while(q>0)
{
  q = q-y;
  y = y+1;
}
```

$$f_1(q,y) = 1 - y, f_2(q,y) = q + 1, C_1 = C_2 = C_3 = 1;$$

Example

Template

$$f_1(q,y) = a_1^T \cdot U_1(q,y) = (a_{11}, a_{12}, a_{13}) \cdot (q, y, 1)$$
$$f_2(q,y) = a_2^T \cdot U_2(q,y) = (a_{21}, a_{22}, a_{23}) \cdot (q, y, 1)$$

Constraint

$$\forall((q,y),(q,y)') \in \Omega : \begin{cases} a_1^T \cdot (U_1(q,y) - U_1(q',y')) \geq C_1 \\ a_2^T \cdot (U_2(q,y) - U_2(q',y')) + a_1^T \cdot U_1(q,y) \geq C_2 \\ a_2^T \cdot U_2(q,y) \geq C_3 \end{cases}$$

Substitute

$$
\begin{aligned}
G_1(q,y,q',y') &\mapsto \begin{pmatrix} 0 \\ U_1(q,y) - U_1(q',y') \end{pmatrix}, \\
G_2(q,y,q',y') &\mapsto \begin{pmatrix} U_2(q,y) - U_2(q',y') \\ U_1(q,y) \end{pmatrix}, \\
G_3(q,y,q',y') &\mapsto \begin{pmatrix} U_2(q,y) \\ 0 \end{pmatrix}
\end{aligned}
$$

Constraint

$$\forall(q,y,q',y') \in \Omega : \begin{cases} (a_2^T, a_1^T) \cdot G_1(q,y,q',y') \geq C_1 \\ (a_2^T, a_1^T) \cdot G_2(q,y,q',y') \geq C_2 \\ (a_2^T, a_1^T) \cdot G_3(q,y,q',y') \geq C_3 \end{cases}$$

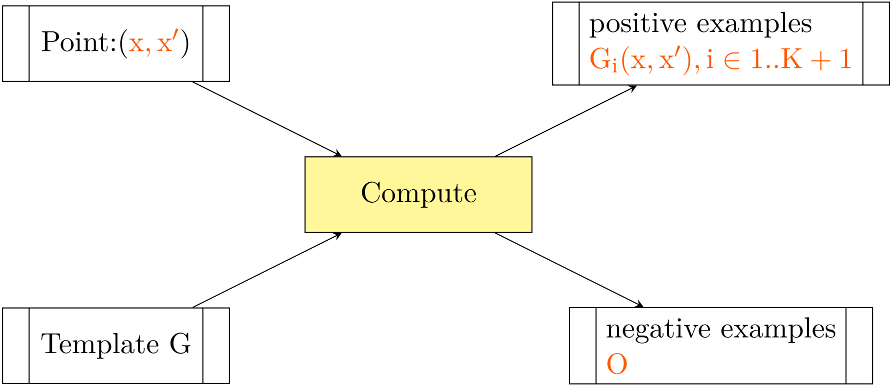
where $C_i > 0, i \in 1..3$.

SVM

$$\forall(q, y, q', y') \in \Omega : \begin{cases} w \cdot G_1(q, y, q', y') + b \geq C_1 > 0 \\ w \cdot G_2(q, y, q', y') + b \geq C_2 > 0 \\ w \cdot G_3(q, y, q', y') + b \geq C_3 > 0 \end{cases}$$

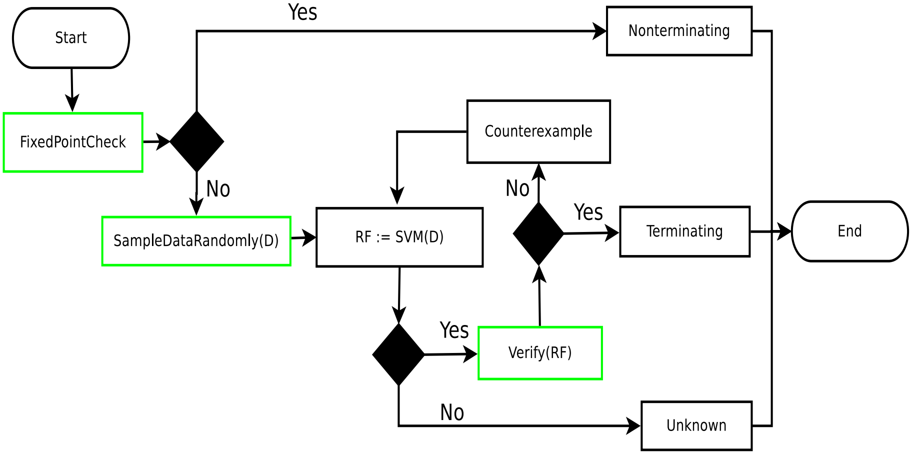
where $w = (a_2^T, a_1^T), b = 0.$

Data set



Point:(x, x')
positive examples
$G_i(x, x'), i \in 1..K+1$
Compute
Template G
negative examples
O

Algorithm



Start
FixedPointCheck
Yes
Nonterminating
No
SampleDataRandomly(D)
RF := SVM(D)
Counterexample
No
Yes
Terminating
End
Yes
Verify(RF)
No
Unknown

Algorithm

Algorithm 1: The SVM-based algorithm for synthesizing k-nested ranking functions

Input: Program Ω, initial sample size n, functions template $U = \langle U_1, \ldots, U_k \rangle$
Output: The k-nested ranking coefficients $(\mathbf{a}_k^T, \ldots, \mathbf{a}_1^T)$ if Ω is well-founded; "nonterminating" if Ω contains fixed points; "unknown" otherwise

begin
  if HasFixedPoint(Ω) **then**
    return (*nonterminating*, GetFixedPoint(Ω));
  $D := \{(\mathbf{O}, -1)\}$;
  for $i := 1$ **to** n **do**
    Sample $(\mathbf{x}, \mathbf{x}')$ randomly from Ω;
    $D := D \cup$ GetDataPoint(U, $(\mathbf{x}, \mathbf{x}')$);
  while *true* **do**
    $svm :=$ SVM(D);
    if $svm = (\mathbf{a}_k^T, \ldots, \mathbf{a}_1^T)$ **then**
      $check :=$ Verify(svm, U, Ω);
      if $check = true$ **then**
        return (*terminating*, *svm*);
      else // *check* is a counterexample of the form $(\mathbf{x}, \mathbf{x}')$
        $D := D \cup$ GetDataPoint(U, *check*);
    else // SVM failed to separate $\mathbf{O}$ from the points from $G(\Omega)$
      return *unknown*

Experiment

- Implement algorithm in a prototype tool named SVMRanker.
- Compare with LassoRanker, which is the main part of the Ultimate Automizer for proving termination of loop programs.
- Linear program
  - All program cases are adapted from the programs in the official website of LassoRanker.
- Non-linear program
  - All cases are adapted from the programs in related papers which focus on the termination of non-linear programs.

Result

	Terminating	Non-terminating	Unknown	Timeout
Dataset	65	69	-	-
SVMRanker	40	34	0	60
LassoRanker	24	37	73	0
Common cases	19	34	0	0

Tab.: Summary of the experiments

Conclusion & Future Work

- Conclusion
  - Based on SVM to synthesize nested ranking functions.
    - Able to deal with the polynomial programs with nested ranking functions.
    - Show the relation between the nested ranking functions and the SVM algorithm.
  - Provide the comprehensive empirical evaluation.
    - Implement algorithm in a prototype tool named SVMRanker.
    - Solve many cases with the non-linear ranking function.
- Future Work
  - Try to apply the learning algorithm to prove the non-termination of programs.
  - Find more efficient SMT tools to verify the ranking function.

Experience of Master Student

- Yi Li, Xuechao Sun, Yong Li, Andrea Turrini, Lijun Zhang: Synthesizing Nested Ranking Functions for Loop Programs via SVM. ICFEM 2019: 438-454.(CCF- C **类会议**, **大会报告作者**)
- Yong Li, Xuechao Sun, Andrea Turrini, Yu-Fang Chen, Junnan Xu: ROLL 1.0: -Regular Language Learning Library. TACAS (1) 2019: 365-371.(CCF-B **类会议**, **工具短文**, **大会报告作者**)

Thanks! & Questions?