Bound Loop

Peisen YAO and Pritom Rajkhowa

Dec 2019

As motivating examples for this research work, we can cite two different programs to demonstrate how the proposed approach works and addresses the limitation of the existing state of the art approaches. These motivating examples provide an intuitive description of three important aspects of our approach. Firstly, the newly proposed approach used Lin's translation to derive the loop bound. Secondly, we demonstrate how our approach successfully finds the loop bound programs with the non-linear loops body, nested loop, multi-path loops body, and loop body with the non-deterministic assignment. It is important to mention that the handling of such programs is one of the major limitations of existing approaches. Some of the approaches adopt the strategy to compute accurate information about syntactically restricted loops. But these strategies are not suitable to handle these programs exposes there limitations. Our new proposed approach addresses all these issues. Lastly, our approach provides efficient methods to synthesize loop bound with such as $O(n \times logn)$ or $O(n^r)$, where r is not an integer.

```
  i= A;
  while (i>1){
     k=1;
     j=1;
     while(j<=n)
     {
        k++;
        j+=k;
     }
     i= i/2;
}
```

Here is the C code snippet P_1 with variables $\vec{X} = \{A, i, j, k\}$. The translation outlined in [1] would generate the following axioms $\Pi_P^{\vec{X}}$:

$$
\begin{aligned}
&A_1 = A, i_1 = i_6(N_2), k_1 = k_6(N_2), j_1 = j_6(N_2),\\
&k_2((n_1+1), n_2) = (k_2(n_1, n_2) + 1),\\
&j_2((n_1+1), n_2) = (j_2(n_1, n_2) + (k_2(n_1, n_2) + 1)),\\
&k_2(0, n_2) = 1, j_2(0, n_2) = 1,\\
&\neg(j_2(N_1(n_2), n_2) \leq A),\\
&(n_1 < N_1(n_2)) \rightarrow (j_2(n_1, n_2) \leq A),\\
&i_6((n_2+1)) = (i_6(n_2)/2),\\
&k_6((n_2+1)) = k_2(N_1(n_2), n_2),\\
&j_6((n_2+1)) = j_2(N_1(n_2), n_2),\\
&i_6(0) = A, k_6(0) = 0, j_6(0) = 0,\\
&\neg(i_6(N_2) > 0), (n_2 < N_2) \rightarrow (i_6(n_2) > 0)
\end{aligned}
$$

Then we passed $\Pi_P^{\vec{X}}$ to cost calculating algorithm which returns following cost equation :

$$
\begin{aligned}
Cost_{\Pi_P^{\vec{X}}} = 1 + (N_2 - 1) + N_2 \times (1 + 1 + (N_1(n_2) - 1) +\\
N_1(n_2) \times (2 + 2) + 2)
\end{aligned}
$$

With some simplification (including finding the closed recurrence relations generated during the translation), the resulted in a set of the following axioms $\Pi_P^{\vec{X}}$:

$$
\begin{aligned}
&A_1 = A, i_1 = A \times 2^{(-N_2)}, k_1 = N_1(N_2 - 1) + 1,\\
&j_1 = ((N_1(N_2 - 1) + 1) \times (N_1(n_2) + 1)/2),\\
&\neg((N_1(n_2) + 1) \times (N_1(n_2) + 1)/2) \leq A),\\
&(n_1 < N_1(n_2)) \rightarrow (((N_1(n_2) + 1) \times (N_1(n_2) + 1)/2) \leq A),\\
&\neg(A^{(-N_2)} > 1), (n_2 < N_2) \rightarrow (A^{(-n_2)} > 1).
\end{aligned}
$$

where A denotes the input of the program variable `A`, A_1 is the output. Similar conventions apply to variables `i`, `j` and `k`. N_2 is a system-generated natural number constant denoting the number of iterations that the outer loop runs before exiting. N_1 is a system-generated natural number function

denoting that for each k, $N_1(k)$ is the number of iterations that the inner loop runs during the k^{th} iteration of the outer loop.

Now $BOUNDFIND$ select $\neg(A \times 2^{(-N_2)} > 1)$ and updated the equation of the form $N_2 == log_2 A$ from the following axioms

$$\neg(A^{(-N_2)} > 1)$$

$BOUNDFIND$ successfuly derive loop bound of the outer most loop $\mathcal{B}^{N_2} = log_2 A$. Similarly from inner most loop $BOUNDFIND$ select $(((N_1(n_2) - 1 + 1) \times (N_1(n_2) - 1 + 1)/2) \leq A)$ and updated the equation of the form $N_1(n_2) \times N_1(n_2)/2 - A == 0$ from the following axioms

$$(n_1 < N_1(n_2)) \rightarrow (((N_1(n_2) + 1) \times (N_1(n_2) + 1)/2) \leq A)$$

The β $(N_1(n_2) = [\sqrt{2 \times A}, \sqrt{-2 \times A}]$ are root of $N_1(n_2) \times N_1(n_2)/2 - A == 0$ find by $EQSOLV$. By considering real and postive root, $BOUNDFIND$ successfuly derive loop bound of the inner most loop $\mathcal{B}^{\forall n_2 . N_1(n_2)} = \sqrt{2 \times A}$. Hence, total loop bound can be calculated as follows:

$$\mathcal{B} = \mathcal{B}^{N_2} \times \mathcal{B}^{\forall n_2 . N_1(n_2)} = log_2 A \times \sqrt{2 \times A}$$

Now substituting bound values in corresponding places in cost equation results following:

$$Cost_{\Pi_P^{\vec{X}}} = 1 + (log_2 A - 1) + log_2 A \times (1 + 1 + (\sqrt{2 \times A} - 1) + \sqrt{2 \times A} \times (2 + 2) + 2)$$

```
int A, i = A, k = 0, j = 0;
while (i > 1)
{
     j = i;
     k = 0;
     while (j <= A)
     {
         k = 0;
         while (k < A)
           {
              k = k + 2;
           }
```

```
        j = j * 2;
      }
    i = i / 2;
}
```

Here is the C code snippet P_1 with variables $\vec{X} = \{A, i, j, k\}$. The translation outlined in [1] would generate the following axioms $\Pi_P^{\vec{X}}$:

$$
\begin{aligned}
&A_1 = A, i_1 = i_9(N_3), k_1 = k_9(N_3), j_1 = j_9(N_3),\\
&k_2((n_1+1), n_2, n_3) = (k_2(n_1, n_2, n_3) + 2),\\
&k_2(0, n_2, n_3) = 0,\\
&\neg(k_2(N_1(n_2, n_3), n_2, n_3) < A)\\
&(n1_< N_1(n_2, n_3)) \rightarrow (k_2(n_1, n_2, n_3) < A),\\
&k_5((n_2+1), n_3) = k_2(N_1(n_2, n_3), n_2, n_3),\\
&j_5((n_2+1), n_3) = (j_5(n_2, n_3) * 2),\\
&k_5(0, n_3) = 0, j_5(0, n_3) = i_9(n_3)\\
&\neg(j_5(N_2(n_3), n_3) \leq A)\\
&(n_2 < N_2(n_3)) \rightarrow (j_5(n_2, n_3) \leq A)\\
&i_9((n_3+1)) = (i_9(n_3)/2)\\
&k_9((n_3+1)) = k_5(N_2(n_3), n_3)\\
&j_9((n_3+1)) = j_5(N_2(n_3), n_3)\\
&i_9(0) = A, k_9(0) = 0, j_9(0) = 0,\\
&\neg(((i_9(N_3)) + 1) > 0)\\
&(n_3 < N_3) \rightarrow (i_9(n_3) > 1)
\end{aligned}
$$

Then we passed $\Pi_P^{\vec{X}}$ to cost calculating algorithm which returns following cost equation :

$$
\begin{aligned}
Cost_{\Pi_P^{\vec{X}}} = 1 + 1 + 1 + (N_3 + 1) + N_3 \times (1 + 1 + (N_2(n_3) + 1) + N_2(n_3)\\
\times(1 + (N_1(n_2, n_3) + 1) + N_1(n_2, n_3) \times (2) + 2) + 2)
\end{aligned}
$$

With some simplification (including finding the closed recurrence relations generated during the translation), the translation outlined in [1] would

generate the following axioms:

$$A_1 = A, i_1 = A \times 2^{-N_3}, k_1 = N_1(N_2(N_3 - 1) - 1, N_3 - 1),$$
$$j_1 = A \times 2^{N_3 - 1} \times 2^{N_2(N_3 - 1)},$$
$$\neg(N_1(n_2, n_3) < A), (n_1 < N_1(n_2, n_3)) \rightarrow n_1 < A)$$
$$\neg(2^{-n_3} \times 2^{N_2(n_3)} \leq A)$$
$$(n_2 < N_2(n_3)) \rightarrow (2^{-n_3} \times 2^{n_2} \leq A)$$
$$\neg(A \times 2^{-N_3} > 1)$$
$$(n_3 < N_3) \rightarrow (A \times 2^{-n_3} > 1)$$

where A denotes the input of the program variable `a`, A_1 is the output. Similar conventions apply to variables `i`, `j` and `k`. N_2 is a system-generated natural number constant denoting the number of iterations that the outer loop runs before exiting. N_3 is a system-generated natural number function denoting that for each k, $N_2(k)$ is the number of iterations that the inner loop runs during the k^{th} iteration of the outer loop. $N_1(k, l)$ is the number of iterations that the innermost loop runs during the k^{th} iteration of the inner loop and the l^{th} iteration of the outer loop.

Now $BOUNDFIND$ select $\neg(A \times 2^{-N_3} > 1)$ and updated the equation of the form $N_3 == log_2 A$ from the following axioms:

$$\neg(A \times 2^{-N_3} > 1)$$

$BOUNDFIND$ successfuly derive loop bound of the outer most loop $\mathcal{B}^{N_3} = log_2 A$. Similarly from first inner most loop $BOUNDFIND$ select $(2^{-n_3} \times 2^{N_2(n_3)-1} \leq A)$ and updated the equation of the form $(2^{-n_3} \times 2^{N_2(n_3)-1} \leq A)$ from the following axiom:

$$(n_2 < N_2(n_3)) \rightarrow (2^{-n_3} \times 2^{n_2} \leq A)$$

The $BOUNDFIND$ successfuly derive loop bound of the first inner loop $\mathcal{B}^{\forall n_3 . N_2(n_3)} = log_2 A + n_3$. Similarly, $BOUNDFIND$ successfuly derive loop bound of the inner most loop $\mathcal{B}^{\forall n_2, n_3 . N_1(n_2, n_3)} = A$ from the following axiom:

$$\neg(N_1(n_2, n_3) < A)$$

Hence, total loop bound can be calculated as follows:

$$\mathcal{B} = \mathcal{B}^{N_1} \times \mathcal{B}^{\forall n_3 . N_2(n_3)} \times \mathcal{B}^{\forall n_2, n_3 . N_1(n_2, n_3)} = (log_2 A) \times (log_2 A + n_3) \times A$$

Need to workout how to resolve loop counter varibles n_2, n_2, n_3. Asymptotic presentation need to do.

Now substituting bound values in corresponding places in cost equation results following:

$$Cost_{\Pi_P^{\vec{X}}} = 1 + 1 + 1 + (log_2 A + 1) + log_2 A \times (1 + 1 + (log_2 A + n_3 + 1) \\ + log_2 A + n_3 \times (1 + (A + 1) + A \times (2) + 2) + 2)$$

To illustrate how our system works, consider the following program for computing the integer square root of a given non-negative integer X:

```
a=0;
su=1;
t=1;
assume(X>=0);
while ( su <= X ) {
  a=a+1;
  t=t+2;
  su=su+t;
}
```

0.1 Goal

The goal is to find the loop bound in terms of input value X with respect to pre-condition $X >= 0$.

0.2 Our Approach To Find Bound

With some simplification, the translation outlined in [1] would generate the following axioms:

$$X_1 = X,$$
$$a_1 = a_3(N), \quad a_3(0) = 0,$$
$$a_3(n+1) = a_3(n) + 1,$$
$$t_1 = t_3(N), \quad t_3(0) = 1,$$
$$t_3(n+1) = t_3(n) + 2,$$
$$su_1 = su_3(N), \quad su_3(0) = 1,$$
$$su_3(n+1) = su_3(n) + t_3(n) + 2,$$
$$\neg(su_3(N) \leq X),$$
$$(n < N) \rightarrow (su_3(n) \leq X),$$

where a denotes the input of the program variable `a`, a_1 is the output, and $a_3(n)$ is a temporary function denoting the value of `a` after the nth iteration of the while loop. Similar conventions apply to X, t, su and their subscripted versions. The constant N is introduced to denote the number of iterations of the while loop before it exits. The variable n ranges over natural numbers and is universally quantified.

By using RS, our translator computes closed-form solutions for $a_3()$, $t_3()$, and $su_3()$, eliminates them, and produces the following axioms:

$$
\begin{aligned}
&X_1 = X, \quad a_1 = N, \quad t_1 = 2N + 1, \\
&su_1 = N^2 + 2N + 1, \\
&(N^2 + 2N + 1 > X), \\
&(n < N) \rightarrow (n^2 + 2n + 1 \leq X).
\end{aligned}
$$

Now $BOUNDFIND$ select $(N-1)^2+2(N-1)+1-X \leq 0$ and updated the equation of the form $(N-1)^2 + 2(N-1) + 1 - X == 0$. The β $(N = [\frac{1}{2} \times (\sqrt{4X+1}-1), \frac{1}{2} \times (\sqrt{4X+1}+1))]$ is root of $N+1-1-X == 0$ find by $EQSOLV$. $BOUNDFIND$ add β in $\mathcal{B}$ and return $\mathcal{B}$.

References

[1] F. Lin, "A formalization of programs in first-order logic with a discrete linear order," *Artificial Intelligence*, vol. 235, pp. 1 – 25, 2016. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S000437021630011X