编译原理期末项目：变量范围分析

01/05/2018

1 背景介绍

数据流分析(data flow analysis)[1]是一种用于收集程序在不同点的状态信息的技术。这些信息通常被编译器用来优化程序。数据流分析的一个典型的例子就是可到达定义的计算，即通过一个程序的控制流图(control flow graph, CFG)来确定对变量的一次赋值可能传播到程序中的哪些部分。

本次课程项目需要大家实现的数据流分析则是对变量的值的范围进行分析，即value range anlysis。变量范围分析追踪所有程序点上各个变量的范围（取值区间）。范围分析的结果可以被用来进行冗余消除、死代码消除、指令选择等优化，也可以被用来检测缓冲区溢出以提高程序的安全性。范围分析可以通过多种算法实现[2, 3, 4]。

2 问题定义

- 输入：静态单赋值(SSA)形式的函数中间代码、函数各个输入参数的范围
- 输出：函数返回值的范围
- 示例：

源文件：

```
/*
 * file: example.cpp
 * input: a in [0, 100]
 * output: a + k in [100, 200]
 */


int example(int a){
  int k = 0;
  while(k < 100){
        int i = 0;
        int j = k;
        while(i < j){
                i = i + 1;
                j = j - 1;
        }
        k = k + 1;
  }
  return a + k;
}
```

输入：

SSA形式中间代码：

```
   int example(int) (int a)
{
  int j;
  int i;
  int k;
  int D.2214;
  int _11;

  <bb 2>:       # bb -> basic block
  k_4 = 0;
  goto <bb 7>;

  <bb 3>:
  i_5 = 0;
  j_6 = k_1;
  goto <bb 5>;

  <bb 4>:
  i_7 = i_2 + 1;
  j_8 = j_3 + -1;

  <bb 5>:
  # i_2 = PHI <i_5(3), i_7(4)>   PHI function
  # j_3 = PHI <j_6(3), j_8(4)>
  if (i_2 < j_3)
    goto <bb 4>;
  else
    goto <bb 6>;

  <bb 6>:
  k_9 = k_1 + 1;

  <bb 7>:
  # k_1 = PHI <k_4(2), k_9(6)>
  if (k_1 <= 99)
    goto <bb 3>;
  else
    goto <bb 8>;

  <bb 8>:
  _11 = a_10(D) + k_1;

<L6>:
  return _11;

}
```

输出：

return value in [100, 200]

3 评分标准

我们提供10组输入输出以便各位进行算法的测试验证，并保留10组数据。评价时，将对这20组数据进行测试，观察各位的算法得到的输出范围是否精确。提供的10组数据，每通过一组得6分；保留的10组数据，每过一组得4分。总分100分。

4 注意事项

- 程序中涉及的运算仅包括基本的加减乘除、比较操作。
- 程序中涉及的数据类型包含int和float point。
- 需要提交的内容包括源码和安装使用说明文档。
- 如果函数返回值的取值范围为一组离散的值，区间为[a, b]，其中a、b分别为这组值中的最小最大值。
- 不同的数据流分析方法得到的结果可能不同，因此边界范围正确即可。
- 提交截至时间为7.4晚24时之前。邮件至1601111271@pku.edu.cn，注明姓名学号。

5 参考文献

1 《编译原理》（龙书/教科书）第9章机器无关优化中的数据流分析(data flow analysis) ⋆

2 Rodrigues, Raphael Ernani, Victor Hugo Sperle Campos, and Fernando Magno Quintao Pereira. "A fast and low-overhead technique to secure programs against integer overflows." Code Generation and Optimization (CGO), 2013 IEEE/ACM International Symposium on. IEEE, 2013.

3 Harrison, William H. "Compiler analysis of the value ranges for variables." IEEE Transactions on software engineering 3 (1977): 243-250.

4 Wagner, David, et al. "A First Step Towards Automated Detection of Buffer Overrun Vulnerabilities." NDSS. 2000.