Liveness-based Reaching Definition Analysis using PRISM

by

Rashmi Rekha Mech
Roll No : 133050089

Under the Guidance of

Prof. Uday Khedker

Abstract

This document provides an updates on implemented liveness-based interprocedural reaching definition analysis. It also provides some data (SPEC Benchmark) to compare with and without liveness-based reaching definition analysis. It also presents some issues in doing pointer analysis.

Description

In liveness-based reaching definition analysis, a definition $d_x : x = e$ reaches a program point u if it appears (without a redefinition of x) on some path from program entry to u and x is live at u. Context sensitive analysis results in more precise information, therefore for inter-procedural analysis we are using context sensitive inter-procedural analysis implemented by Vinit.

Data flow equations for Liveness-based reaching definition analysis :

$$LIn_n = f_n(Out_n)$$

$$LOut_n = \begin{cases} BI & \text{n is End} \\ \bigcup\limits_{s \in succ(n)} In_s & \text{otherwise} \end{cases}$$

where,

$$f_n(X) = \begin{cases} (X - \{y\}) \cup (\mathit{Opd(e)} \cap \mathit{Var}) & \text{n is y = e, e} \in \text{Expr, y} \in \text{X} \\ X - y & \text{n is input(y)} \\ X \cup y & \text{n is use(y)} \\ X & \text{otherwise} \end{cases}$$

$$RIn_n = \begin{cases} RBI & \text{n is Start block} \\ \bigcup\limits_{p \in pred(n)} Out_p \mid_{LIn_n} & \text{otherwise} \end{cases}$$

$$ROut_n = Gen_n \cup (In_n - Kill_n) \mid_{LOut_n}$$

$$RBI = \{d_x : x = \mathit{undef} \mid x \in \mathit{Var}\} \tag{1}$$

SPEC Benchmark Evaluation

In order to compare the performance between two analysis i.e. inter-procedural reaching definition analysis with and without liveness, we tested both for SPEC Benchmarks. For each query, we measured the average size of the set of data flow values computed at each program point. In general, average size of the set at each program point in liveness-based analysis is much smaller than that of the normal reaching definition analysis.

Performance measurement of reaching definition without liveness is shown in table 1.

Name	No. of Basic Blocks	Avg. values at Entry	Avg. values at Exit	Total values
bzip2	8004	113.89	114.18	114.04
mcf	1066	94.90	95.01	94.96
hmmer	24996	57.53	57.70	57.61
sjeng	10892	39.219	39.245	39.232
lbm	603	10.53	10.71	10.62

Table 1: Benchmark results for normal reaching definition analysis.

The performance measurement of reaching definition with liveness is shown in table 2.

Name	No. of Basic Blocks	Avg. values at Entry	Avg. values at Exit	Total values
bzip2	8004	10.45	10.23	10.34
mcf	1066	22.07	21.77	21.92
hmmer	24996	1.38	1.34	1.36
sjeng	10892	11.96	11.69	11.82
lbm	603	1.50	1.46	1.48

Table 2: Benchmark results for liveness-based reaching definition analysis.

Figure 1 shows a comparision between two analysis.

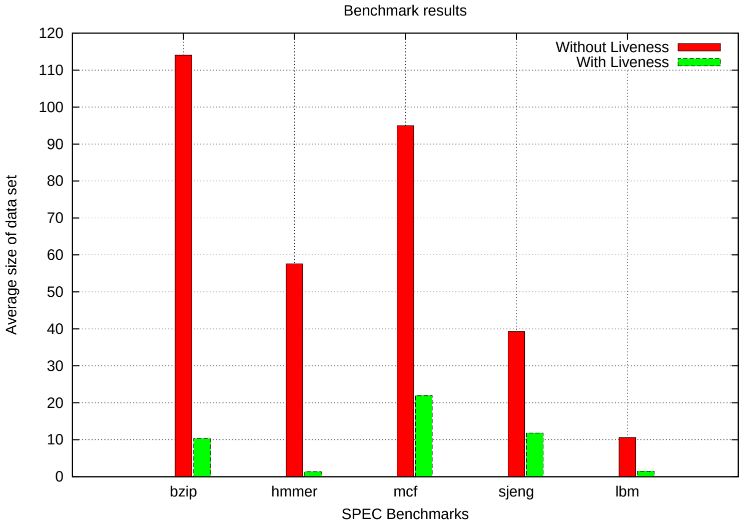

Figure 1: Percentage reduction in size of data set for Reaching definition analysis with and without liveness

Difficulties

We need pointer information for expressions containing pointer variables but due to some changes in solver, the old alias query produces an empty result. To solve this problem, we are computing a set of pointer information at each program point by using Vinit's pointer analysis query. Also, we are only able to run five SPEC Benchmarks as shown in the tables 1 and 2. We are not able to run the following Benchmarks :

- 464.h264ref : It is taking a long time to execute. Large number of pointers availble in this Benchmark may be the reason for taking long time to execute.
- 403.gcc : It gives error in opening "sbitmap.st" file. This file is not generated during IR generation.
- 462.libquantum : This is also giving the same error "error in opening shor.st file ".
- 433.milc : This is also giving the same error "error in opening control.st file ".