

Improving Program Slicing with Dynamic Points-To Data

Darren Atkinson

Department of Computer Engineering
Santa Clara University
2002

ABSTRACT

Problem
statement

Program slicing is a potentially useful analysis for aiding program understanding. However, slices of even small programs are often too large to be generally useful. Imprecise pointer analyses have been suggested as one cause of this problem. In this paper, we use dynamic points-to data, which represents optimal or optimistic pointer information, to obtain a bound on the best case slice size improvement that can be achieved with improved pointer precision. Our experiments show that slice size can be reduced significantly for programs that make frequent use of calls through function pointers because for them the dynamic pointer data results in a considerably smaller call graph, which leads to fewer data dependences. Programs without or with only few calls through function pointers, however, show only insignificant improvement. We identified Amdahl's law as the reason for this behavior: C programs appear to have a large fraction of direct data dependencies so that reducing spurious dependencies via pointers is only of limited benefit. Consequently, to make slicing useful in general for such programs, improvements beyond better pointer analyses will be necessary. On the other hand, since we show that collecting dynamic function pointer information can be performed with little overhead (average slowdown of 10% for our benchmarks), dynamic pointer information may be a practical approach to making slicing of programs with frequent function pointer use more successful in reality.

Work
Accomplished

Results

Our experiments show that slice size can be reduced significantly for programs that make frequent use of calls through function pointers because for them the dynamic pointer data results in a considerably smaller call graph, which leads to fewer data dependences. Programs without or with only few calls through function pointers, however, show only insignificant improvement. We identified Amdahl's law as the reason for this behavior: C programs appear to have a large fraction of direct data dependencies so that reducing spurious dependencies via pointers is only of limited benefit. Consequently, to make slicing useful in general for such programs, improvements beyond better pointer analyses will be necessary. On the other hand, since we show that collecting dynamic function pointer information can be performed with little overhead (average slowdown of 10% for our benchmarks), dynamic pointer information may be a practical approach to making slicing of programs with frequent function pointer use more successful in reality.

Conclusions

Recommendations

Consequently, to make slicing useful in general for such programs, improvements beyond better pointer analyses will be necessary. On the other hand, since we show that collecting dynamic function pointer information can be performed with little overhead (average slowdown of 10% for our benchmarks), dynamic pointer information may be a practical approach to making slicing of programs with frequent function pointer use more successful in reality.

Keywords: Program Slicing, Points-To Analysis, Dynamic Analysis.