Available online at www.sciencedirect.com
SCIENCE DIRECT®
Science of Computer Programming 58 (2005) 83–114
Science of Computer Programming
www.elsevier.com/locate/scico

# Error checking with client-driven pointer analysis☆

Samuel Z. Guyer*, Calvin Lin

*The University of Texas at Austin, Department of Computer Sciences, Austin, TX 78712, USA*

Received 20 December 2003; received in revised form 28 May 2004; accepted 17 February 2005
Available online 31 May 2005

**Abstract**

This paper presents a new *client-driven* pointer analysis algorithm that automatically adjusts its precision in response to the needs of client analyses. Using five significant error detection problems as clients, we evaluate our algorithm on 18 real C programs. We compare the accuracy and performance of our algorithm against several commonly used fixed-precision algorithms. We find that the client-driven approach effectively balances cost and precision, often producing results as accurate as fixed-precision algorithms that are many times more costly. Our algorithm works because many client problems only need a small amount of extra precision applied to selected portions of each input program.
© 2005 Elsevier B.V. All rights reserved.

*Keywords:* Pointer analysis; Context-sensitive; Flow-sensitive; Adaptive analysis; Error checking; Error detection

## 1. Introduction

Pointer analysis is critical for effectively analyzing programs written in languages such as C, C++, and Java, which make heavy use of pointers and pointer-based data structures. The goal of pointer analysis is to disambiguate indirect memory references so that subsequent compiler passes have a more accurate view of program behavior. In this sense, pointer analysis is not a stand-alone task: its purpose is to provide pointer information to other *client* analyses.

☆ This work is supported by NSF grants CCR-0085792, EIA-0303609, ACI-0313263, ACI-9984660, DARPA Contract #F30602-97-1-0150, and an IBM Faculty Partnership Award.
* Corresponding author. Tel.: +1 512 232 7471.
*E-mail address:* sammy@cs.utexas.edu (S.Z. Guyer).

0167-6423/$ - see front matter © 2005 Elsevier B.V. All rights reserved.
doi:10.1016/j.scico.2005.02.005

```
p = safe_string_copy("Good");      char * safe_string_copy(char * s)
q = safe_string_copy("Bad");       {
r = safe_string_copy("Ugly");        if (s != 0) return strdup(s);
                                     else return 0;
                                   }
```

Fig. 1. Context-insensitive pointer analysis hurts accuracy, but whether or not that matters depends on the client analysis.

Existing pointer analysis algorithms differ considerably in their precision. Previous research has generally agreed that more precise algorithms are often significantly more costly to compute, but previous work has disagreed on whether more precise algorithms yield more accurate results and whether these results are worth the additional cost [30, 28,19,10,26]. In fact, a recent survey by Hind claims that the choice of pointer analysis algorithm should be dictated by the needs of the client analyses [18].

In this paper we present a new *client-driven* pointer analysis algorithm that addresses this viewpoint directly: it automatically adjusts its precision to match the needs of the client. The key idea is to discover where precision is needed by running a fast initial pass of the client. The pointer and client analyses run together in an integrated framework, allowing the client to provide feedback about the quality of the pointer information that it receives. Using these initial results, our algorithm constructs a precision policy customized to the needs of the client and input program. This approach is related to demand-driven analysis [20,17] but solves a different problem: while demand-driven algorithms determine which parts of the analysis need to be computed, client-driven analysis determines which parts need to be computed using more precision.

As an example of how different clients require different amounts of precision, consider a context-insensitive analysis of the string copying routine in Fig. 1: the pointer parameter `s` merges information from all the possible input strings and transfers it to the output string. For a client that associates dataflow facts with string buffers, this could severely hurt accuracy—the appropriate action is to treat the routine context-sensitively. However, for a client that is not concerned with strings, the imprecision is irrelevant.

We evaluate our algorithm using five security and error detection problems as clients. These clients are demanding analysis problems that stress the capabilities of the pointer analyzer, but with adequate pointer analysis support they can detect significant and complex program defects. We compare our algorithm against four fixed-precision algorithms on a suite of 18 real C programs. We measure the cost in terms of time and space, and we measure the client's accuracy simply as the number of errors reported: the analysis is conservative, so fewer error reports always indicates fewer false positives.

This paper, which is an extended version of earlier work [16], makes the following contributions. (1) We present a client-driven pointer analysis algorithm that adapts its precision policy to the needs of client analyses. For our five error detection clients, this algorithm effectively discovers where to apply more analysis effort to reduce the number of false positives. (2) We present empirical evidence that different analysis clients benefit from different kinds of precision—flow sensitivity, context sensitivity, or both. In most cases only a small part of each input program needs such precision; our algorithm works

Download English Version:

https://daneshyari.com/en/article/9657406

Download Persian Version:

https://daneshyari.com/article/9657406

Daneshyari.com