ΠΑΝΕΠΙΣΤΗΜΙΟ ΚΡΗΤΗΣ
ΤΜΗΜΑ ΕΠΙΣΤΗΜΗΣ ΥΠΟΛΟΓΙΣΤΩΝ

ΠΑΡΟΥΣΙΑΣΗ / ΕΞΕΤΑΣΗ ΜΕΤΑΠΤΥΧΙΑΚΗΣ ΕΡΓΑΣΙΑΣ

Σταυρακαντωνάκη Ειρήνη

Μεταπτυχιακή Φοιτήτρια

Τμήμα Επιστήμης Υπολογιστών, Πανεπιστήμιο Κρήτης

Επόπτης Μεταπτ. Εργασίας: Καθηγητής Άγγελος Μπίλας

Δευτέρα, 25/07/2016, 11:00

Αίθουσα Β108, Τμήμα Επιστήμης Υπολογιστών, Πανεπιστήμιο Κρήτης

" 'Μια στατική ανάλυση δεικτών σε ενδιάμεση αναπαράσταση για βελτιστοποιήσεις μεταγλωττιστών "

Η ανάλυση δεικτών είναι σημαντικό συστατικό των περισσότερων στατικών αναλύσεων. Η στατική ανάλυση δεικτών χρίζει πολλών εφαρμογών, όπως η αύξηση της ακρίβειας και της αποδοτικότητας των βελτιστοποιήσεων στους μεταγλωττιστές προγραμμάτων, οι αυτόματες παραλληλοποιήσεις κώδικα, και τα εργαλεία εντοπισμού λαθών. Συνήθως οι υλοποιήσεις ανάλυσης δεικτών βασίζονται σε γλώσσα πηγαίου κώδικα κι εκμεταλλεύονται τα χαρακτηριστικά των γλωσσών για να αυξήσουν την ακρίβεια των απαντήσεων τους, όπως πχ. το σύστημα τύπων. Ωστόσο, αυτό επιτρέπει σε διαφορετικές υλοποιήσεις να δώσουν διαφορετικά αποτελέσματα βάση της ακρίβειας κι της απόδοσης τους, καθιστώντας μ' αυτόν τον τρόπο, απαραίτητη την υλοποίηση διαφορετικού αλγορίθμου για κάθε διαθέσιμο front-end ενός πολυγλωσσικού μεταγλωττιστή. Αντίθετα, η ανάλυση δεικτών για μια γλώσσα ενδιάμεσης αναπαράστασης είναι πιο ευρεία, θυσιάζοντας, όμως, την ακρίβεια που δίνει μια γλώσσα πηγαίου κώδικα. Συνεπώς, βρισκόμαστε μπροστά σ' έναν ενδιαφέρον συμβιβασμό, όπου ο βέλτιστος τρόπος επιτυγχάνει τη μέγιστη ακρίβεια για τη `χαμηλότερη' δυνατή αναπαράσταση γλώσσας.

Η εργασία αυτή παρουσιάζει τη σχεδίαση κι υλοποίηση μιας ανάλυσης δεικτών για τη γλώσσα αναπαράστασης του LLVM. Υλοποιήσαμε μια επέκταση της ανάλυσης δεικτών τύπου Andersen, που χρησιμοποιείται ευρέως για τον υπολογισμό τέτοιας πληροφορίας κυρίως λόγω της κλιμακωσιμότητα (scalability) της. Η ανάλυση υπολογίζεται κατευθείαν στη γλώσσα αναπαράστασης του LLVM, καθιστώντας την εφαρμόσιμη σε όλες τις γλώσσες που μπορούν να μεταφραστούν σ'αυτή την αναπαράσταση. Σε όρους στατικής ανάλυσης, η ανάλυση είναι ένας type-based, inter-procedural, flow-insensitive, και context-insensitive αλγόριθμος, ακριβώς όπως οι περισσότερες εφαρμογές του αλγορίθμου Andersen, με την προσθήκη όμως του field-sensitivity για περισσότερη ακρίβεια σε τύπους δυναμικών δομών. Ο αλγόριθμος είναι σε θέση να ανταποκριθεί σε μια πλήρη σειρά ερωτημάτων δεικτών και να παρέχει σημαντικά μεγαλύτερη ακρίβεια απ' ότι ο

κλασσικός αλγόριθμος, λόγω του field-sensitivity. Αξιολογήθηκε χρησιμοποιώντας 3 διαφορετικές σουίτες από 20 προγράμματα C ανοικτού κώδικα που κυμαίνονται από χίλιες μέχρι 230 χιλιάδες γραμμές πηγαίου κώδικα, αποδεικνύοντας την αποτελεσματικότητα της τεχνικής και παρέχοντας μια ποιοτική αξιολόγηση προς άλλες αλγοριθμικές προσεγγίσεις.

Eirini Stavrakantonaki
M.Sc. Thesis
Computer Science Department
University of Crete
Master's Thesis Supervisor: Professor A. Bilas

Monday, 25/07/2016, 11:00
Room B108, Computer Science dept., University of Crete

"A static pointer analysis on intermediate representation for compilation optimizations"

ABSTRACT

Pointer analysis is a major component of most static program analyses. Static pointer analysis has many uses like increasing the precision and efficiency of compiler optimizations, driving auto-parallelization, and bug-finding tools. Usually pointer analysis implementations target one given source language and take advantage of specific features of that language to increase precision, e.g., the type system, package system, or object inheritance mechanism. This, however, makes each implementation give different results with respect to precision and performance, and often requires a new implementation for each front-end of a multi-language compiler system. Conversely, implementing a pointer analysis for a compiler intermediate representation makes it more generic, while sacrificing precision that could have been achieved by working on the source-language abstraction level. This creates an interesting trade-off, where the optimal point would achieve maximum precision for the lowest-level language that is possible.

This thesis presents the design and implementation of a static pointer analysis for the LLVM intermediate representation language. We implemented an extension of Andersen's pointer analysis algorithm, broadly used to compute such information due to its scalability. The analysis is computed directly from the intermediate representation of LLVM, making it applicable to all languages that compile to the LLVM intermediate representation. The analysis is type-based, inter-procedural, flow-insensitive, and context-insensitive, following most applications of Andersen's algorithm, with the extension of field-sensitivity for increased precision on struct types. Our algorithm is capable of responding to a full variety of alias analysis queries and can provide substantially more precision than the standard algorithm due to its field-sensitivity. Evaluated using three different suites of 20 open-source C programs ranging from 1K to 230K lines of source code, we demonstrate our technique and provide a qualitative evaluation towards other algorithmic approaches.