The follow up of DAEDALUS

Patrick COUSOT
ENS, 45 rue d'Ulm
75230 Paris cedex 05, France
Patrick.Cousot@ens.fr
www.di.ens.fr/ cousot

DAEDALUS industrial seminar, Saarbrücken, 27 Sep. 2002

Problems and Solutions

On resource explosion in program verification

Time/memory resource explosion is frequent in automatic program verification. Solutions:

- Deductive methods: ask the user to better organize the proof and provide inductive arguments;
- Model checking: ask the user to provide a smaller program model;

or

- Use abstraction. Immediate consequence: false alarms due to the imprecision of the approximation.

On the precision of static program analysis

- Applications where imprecision is acceptable:
  - program optimization: do not optimize when unkown,
  - WCET: overestimate time when unkown,
  - debugging: use tests when unkown;
- Applications where precision is mandatory:
  - verification of absence of RTE in safety critical software.

Understanding the origin of (false) alarms

- It may be difficult to understand the origin of (false) alarms;
- Abstract interpretation can help to:
  - Locate the program slice and input data at the origin of the alarm (abstract slicing),
  - Locate the approximations at the origin of the false alarm (analysis monitoring).

Adaptative static analysis: towards no false alarm

- It is theoretically possible to reach zero false alarm by adapting the analysis to a well-defined class of programs [1]:
  - By user-provided abstraction (e.g. model-checking with user provided model), which is too expensive;
  - By adapting the analyzer to the program class specificities (e.g. synchronous programs), as shown by ENS/CNRS-X outside DAEDALUS.

Reference

[1] P. Cousot. *Partial Completeness of Abstract Fixpoint Checking, invited paper*. In *Proc. 4th Int. Symp. SARA '2000*, B.Y. Choueiry and T. Walsh (Eds). Horseshoe Bay, Texas, USA, 26–29 Jul. 2000, LNAI 1864. Springer-Verlag, pp. 1–25, 2000.

Conclusions

Applicability in the near future

- Applications of abstract interpretation to software verification seems feasible in the near future;

THE END, THANK YOU