Abstract Interpretation Based Software Technologies

Patrick COUSOT

École Normale Supérieure
45 rue d'Ulm, 75230 Paris cedex 05, France
Patrick.Cousot@ens.fr
www.di.ens.fr/~cousot

Workshop on Software Technologies, Embedded Systems and Distributed Systems in the sixth Framework Programme
European Commission, Brussels 2 May 2002

What is (or should be) the essential preoccupation of computer scientists?

The production of reliable software, its maintenance and safe evolution year after year (up to 20 even 30 years).

Software Design/Engineering

- A well-organized large international community;
- Lot of progress (OO, components, etc.)
- But:
  - Good for software created ex-nihilo but hard to apply to existing software, its reuse and modification;
  - Basic tools are still traditional compilers, version managers and debuggers;
  - Traditional errors (at interfaces, unexpected exceptions, etc.) still found by manual debugging (which does not scale up).

Formal Methods

- A well-organized large international community;
- Lot of progress (factor of 100 in formal design, deductive methods, model checking, etc. in past 10 years);
- But:
  - Does consider hand-made models not program semantics;
  - Still to scale up (design, maintenance, exhaustive exploration of hand-made models is extremely costly);
  - Does not fulfill their verification promises (debugging).

Approximate Methods

Effective static program analyzes from the **DA∃D∀LUS** project:

- On large critical embedded real-time synchronous avionic software (250 000 lines of C):
  - Absence of unexpected interrupts;
  - WCET: worst-case execution time (after compilation on a given processor);
- On large commercial asynchronous distributed avionic software (80 000 lines of C & POSIX):
  - Presence/absence of race-conditions;
- On excerpts of embedded real-time avionic software:
  - Localization of the origin and estimation of floating-point errors;

Abstract Interpretation Based Methods

- A bit scattered community (European origin);
- A large variety of applications from program compilation, reverse engineering, mobile computing to security of cryptographic protocols, codesign, etc;
- A large variety of languages, from Prolog, C to Java(Card);
- A large variety of subjects from very theoretical to applied experimentations;
- Difficult (real large programs are much harder to handle than small handmade software models);
- At the beginning of industrialization in Europ (AbsInt Angewandte Informatik a, PolySpace)

European Perspectives

- Support the community of European researchers working on abstract interpretation and static analysis;
- Avoid the *pensée unique* and favorize alternative solutions;
- Allow for transversal research (on solutions common to hardware/sofware, embedded/distributed/mobile software, security/correctness verification, etc) as opposed to short-term application domains;
- Favorize the dissemination in education and industry;
- Favorize the international cooperation (Australia, Korea, USA);

Through a Network of Excellence and/or an Integrated Project!