FROM RESEARCH TO INDUSTRY

Combining static & dynamic analysis for software verification

PETIOT Guillaume
PhD student

KOSMATOV Nikolaï
GIORGETTI Alain
Advisors

JULLIAND Jacques
Supervisor

PLAN

SUBJECT

Combining static & dynamic analysis for software verification

- Static analysis: abstract interpretation, deductive verification, etc.
  - ✓ complete
  - ✗ imprecise

- Dynamic analysis: test, runtime verification
  - ✓ precise
  - ✗ incomplete

CONTEXT
SOFTWARE VERIFICATION AT CEA/LSL

Frama-C – framework of modular analysis of C

Kernel

ACSL – specification language

Plug-ins

Value	Value analysis
WP	Deductive verification
PathCrawler	Tests generation, all-paths coverage
E-ACSL	Translation from ACSL to C
SANTE	Collaboration Value + slicing + PathCrawler
Etc.	

PROBLEM, MOTIVATIONS, SOLUTIONS

PROBLEM

- How to verify generic properties like runtime errors not handled by SANTE ?
- And properties specified by the user ?

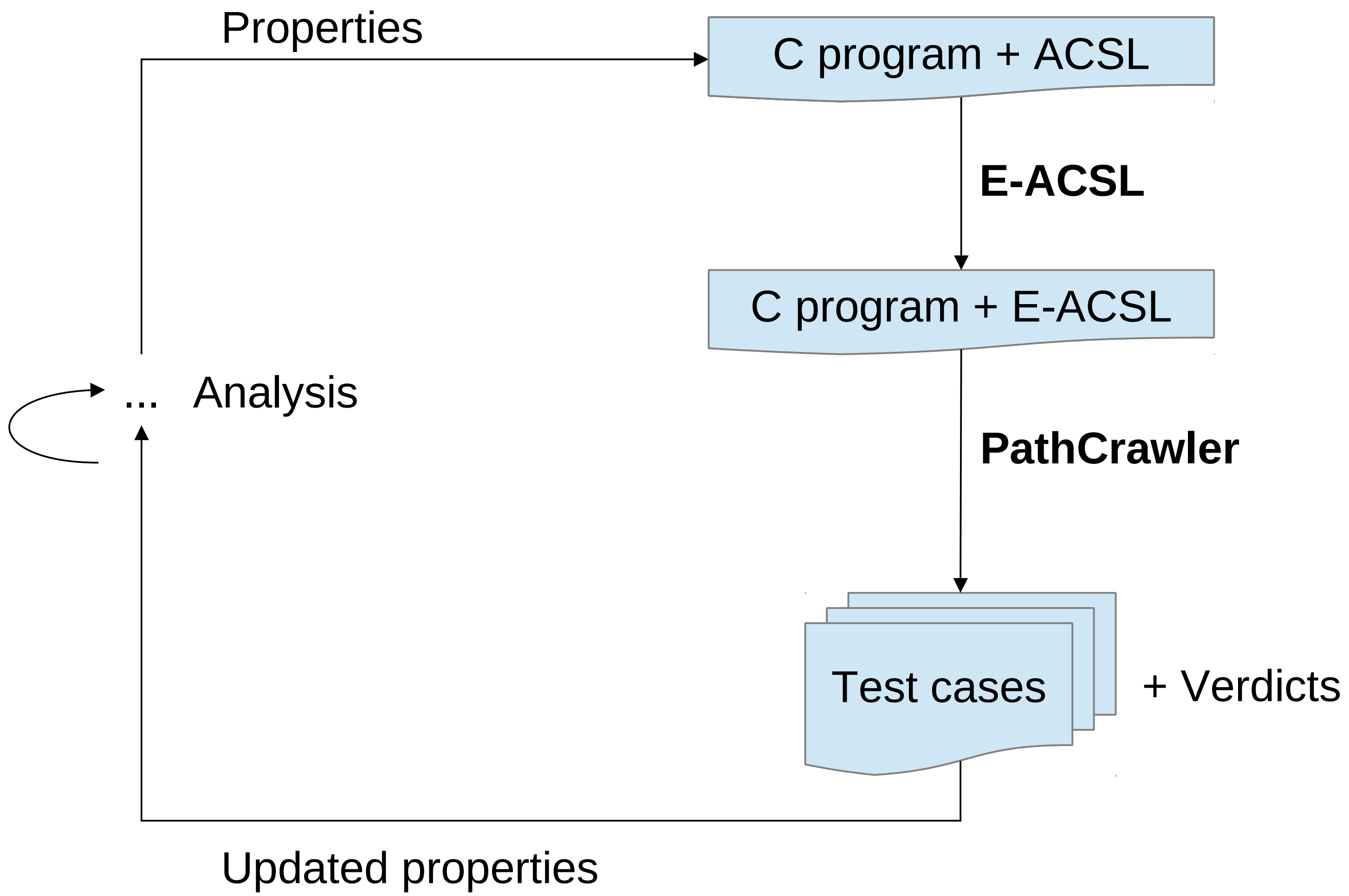
MOTIVATIONS

- Automation
- Reliable alarm classification: low « false positives » rate
- Overcome the drawbacks of each method

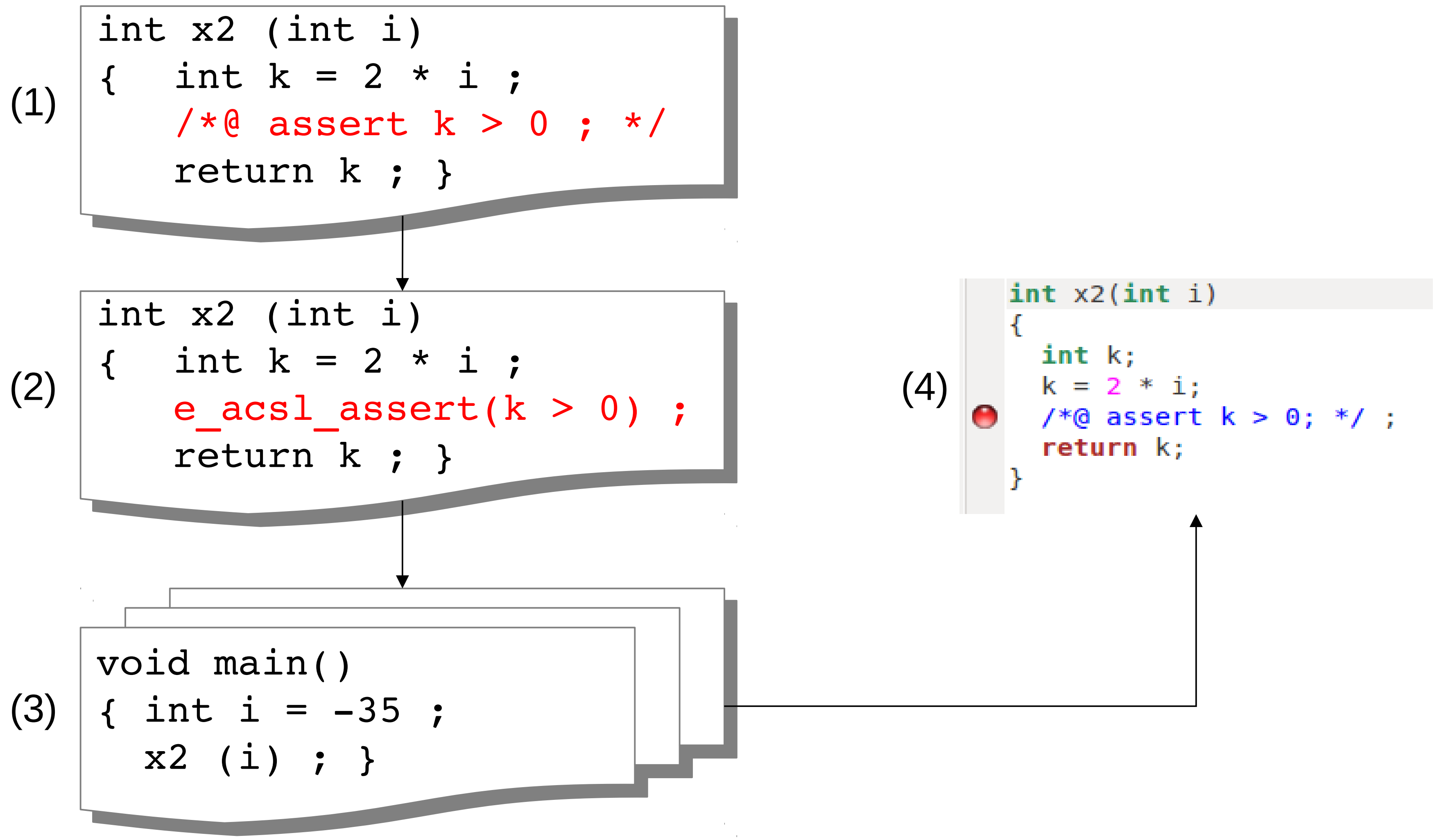
SOLUTIONS

- Combining slicing, static analysis and testing to overcome drawbacks of each method

OVERVIEW


Properties
C program + ACSL
E-ACSL
C program + E-ACSL
PathCrawler
... Analysis
Test cases
+ Verdicts
Updated properties

EXAMPLE



(1)

```
int x2 (int i)
{   int k = 2 * i ;
    /*@ assert k > 0 ; */
    return k ; }
```

(2)

```
int x2 (int i)
{   int k = 2 * i ;
    e_acsl_assert(k > 0) ;
    return k ; }
```

(3)

```
void main()
{ int i = -35 ;
  x2 (i) ; }
```

(4)

```
int x2(int i)
{
  int k;
  k = 2 * i;
  /*@ assert k > 0; */ ;
  return k;
}
```

CONCLUSION, PERSPECTIVES

CONCLUSION

- Handling ACSL annotations by PathCrawler
  - Safety assertions on overflows, pointers, etc.
- Counter-example generation
  - Updating property status in Frama-C
  - Reusable in other plug-ins

PERSPECTIVES

- Handling ACSL pre- and post-condition by PathCrawler
- Application, experiments

Thank you