Extraction-based Regression Test Selection

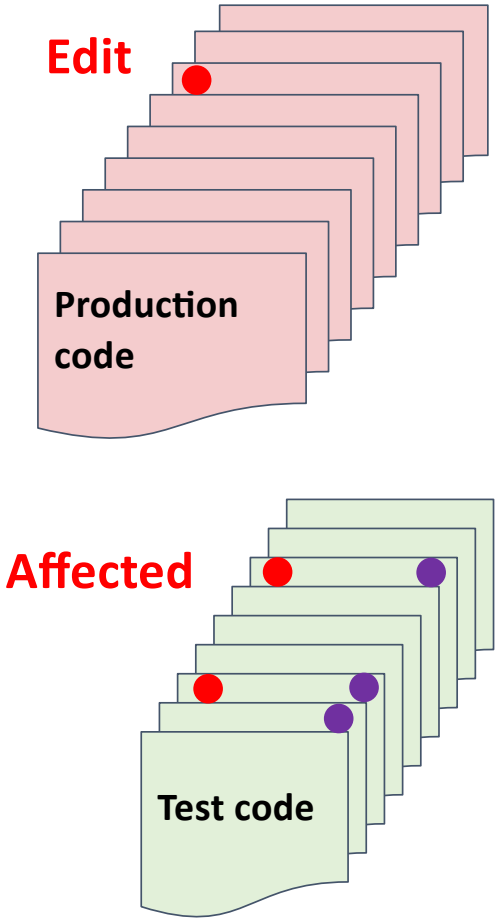
Görel Hedin, Lund University, Sweden

Joint work with Jesper Öqvist and Boris Magnusson

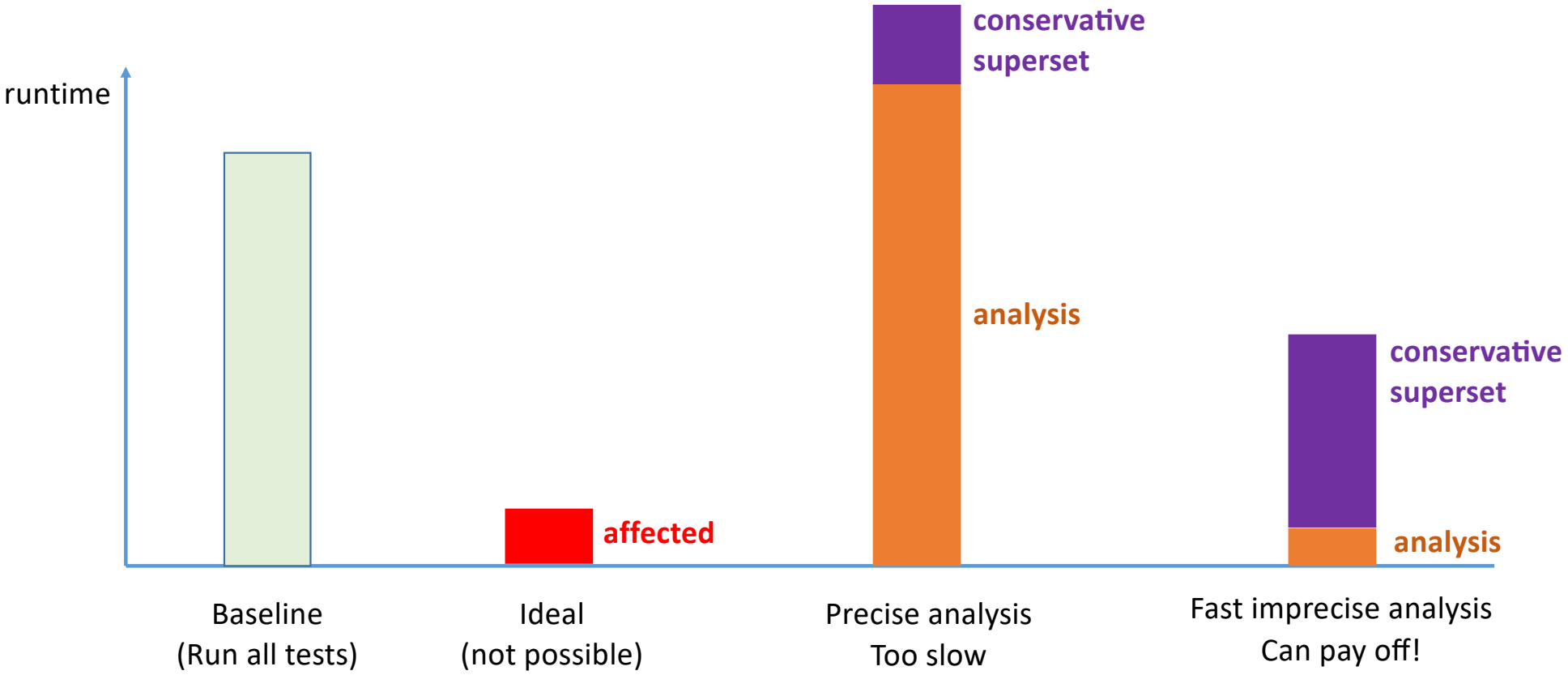
Will also mention subsequent work by
MSc students Filip Stenström and Markus Olsson
and their advisors Niklas Fors and Jon Sten

Regression Testing

- **Regression Testing**
  - Re-run *all tests* (ensure previous functionality still works)
  - Can take a long time...
- **Regression Test Selection (RTS)**
  - Run a subset of all tests
- **Safe RTS (equivalent to Test All)**
  - Run all **affected tests** (changed outcome because of edit)
  - Analysis to find **conservative superset**
  - But **analysis** takes time too...



Does safe RTS pay off?



Practical approaches to safe RTS

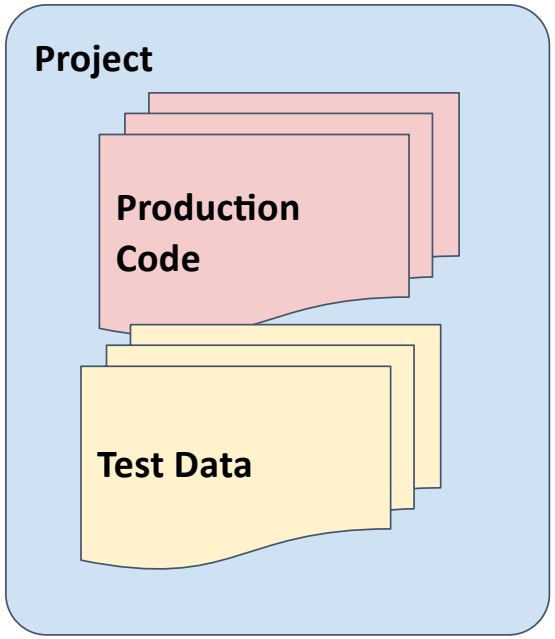
- Work at a coarse-grained level (file/class)

- Dynamic dependency analysis
  - instruments the tests
  - Example tool for Java: Ekstazi 2015

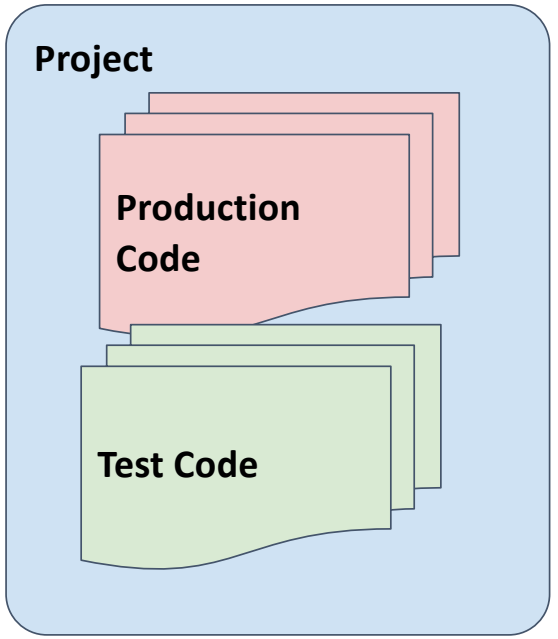
- Static dependency analysis
  - code analysis
  - Example tools for Java: AutoRTS 2016, STARTS 2016
  - Tool for Modelica 2018

MSc project

Input-driven testing



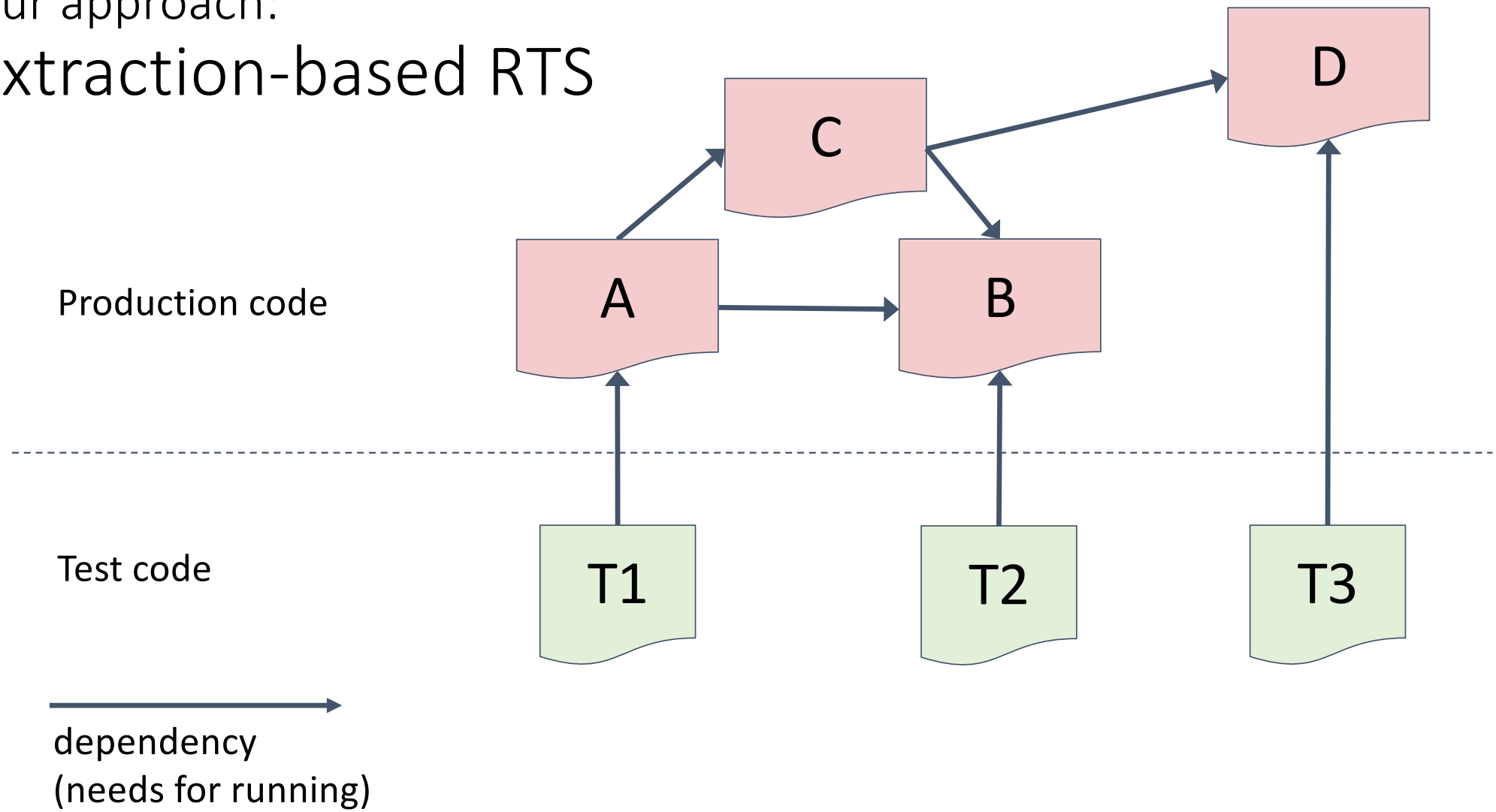
Program-driven testing (xUnit-style)



Our approach

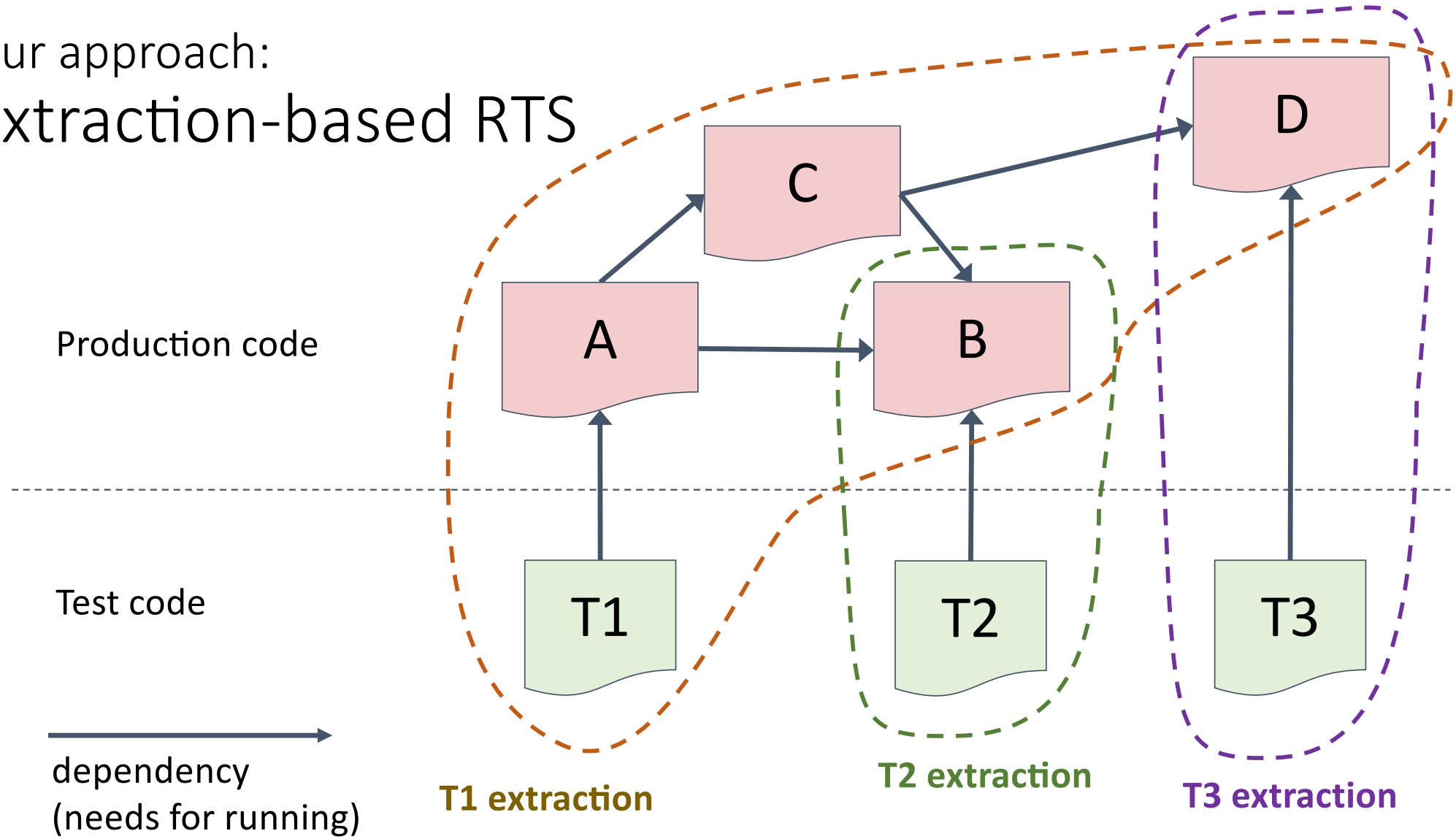
Our approach:

Extraction-based RTS



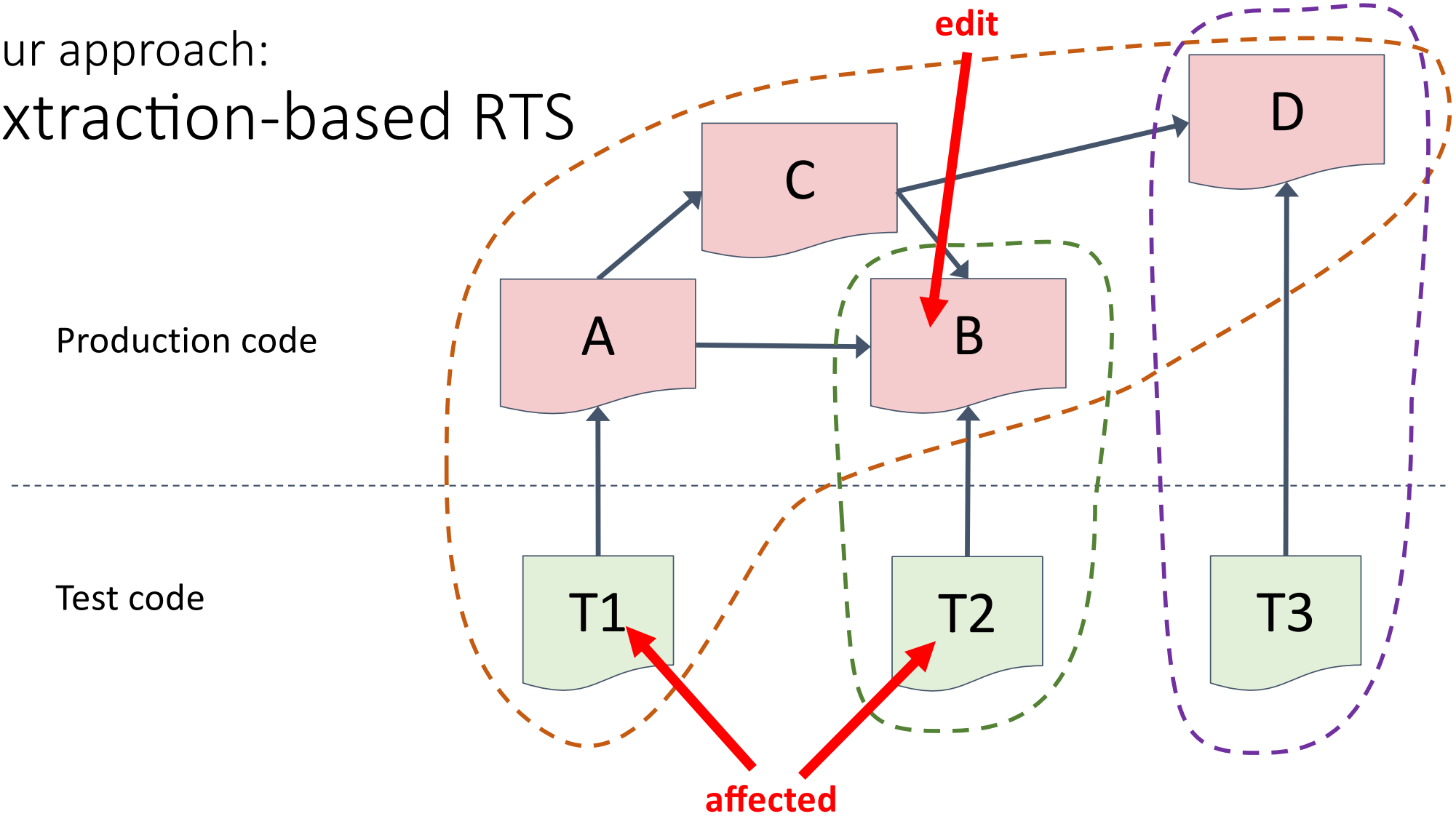
Our approach:

Extraction-based RTS

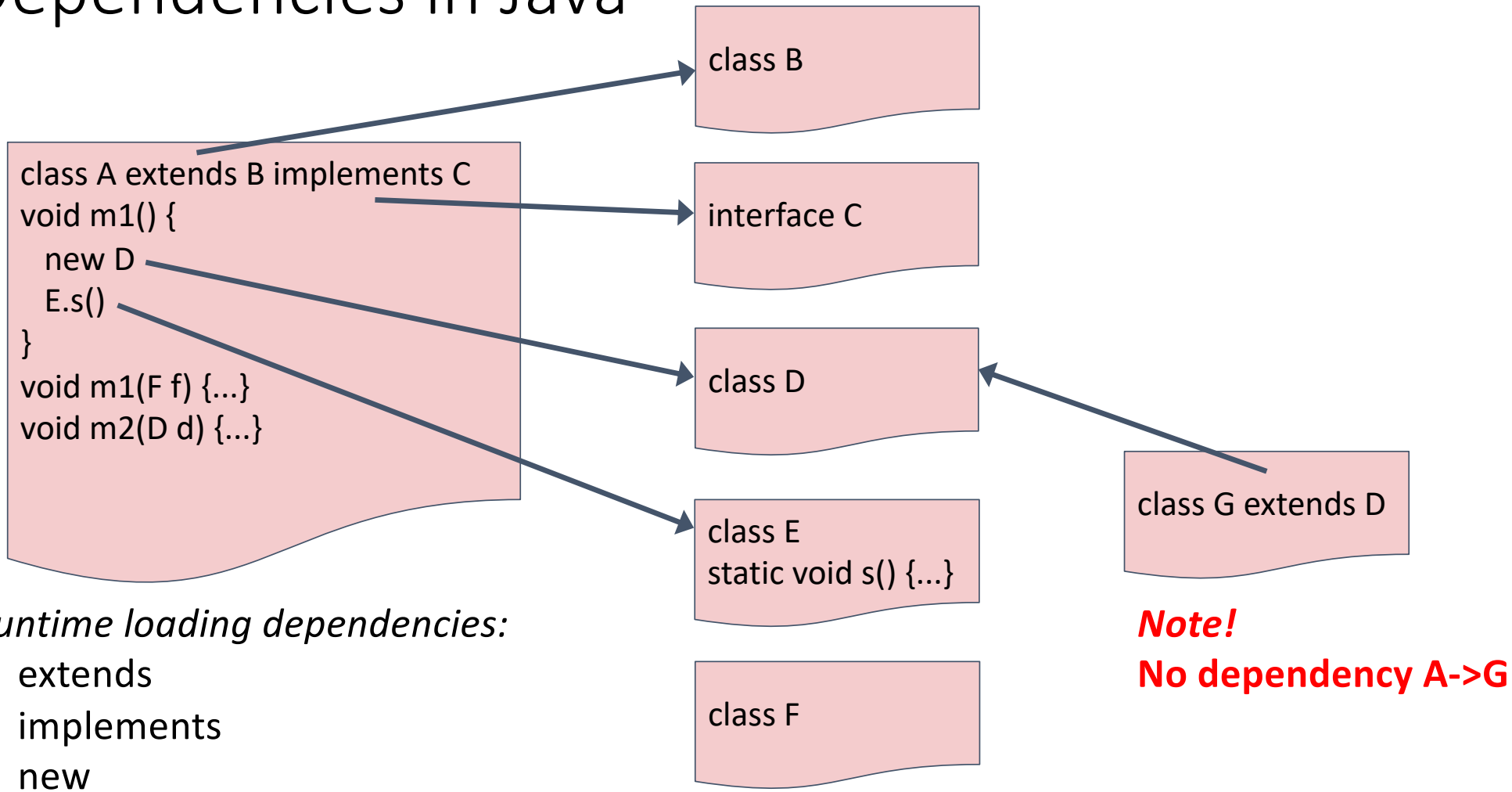


Our approach:

Extraction-based RTS



Dependencies in Java



Runtime loading dependencies:

- extends
- implements
- new
- static

Note!
No dependency A->G

Note! **No dependency A->F**

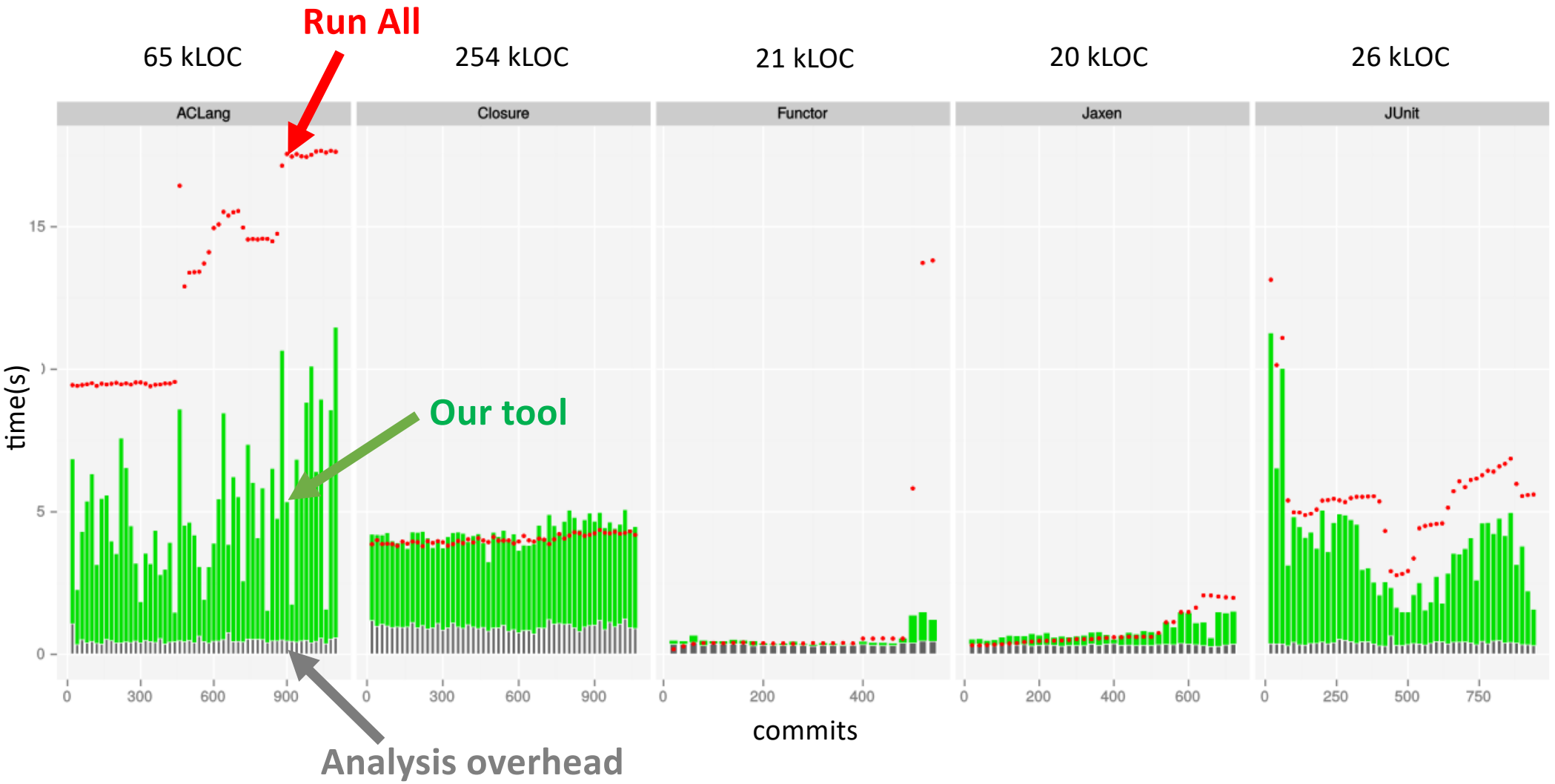
Tool Implementation - AutoRTS

We implemented our analysis for Java and JUnit.

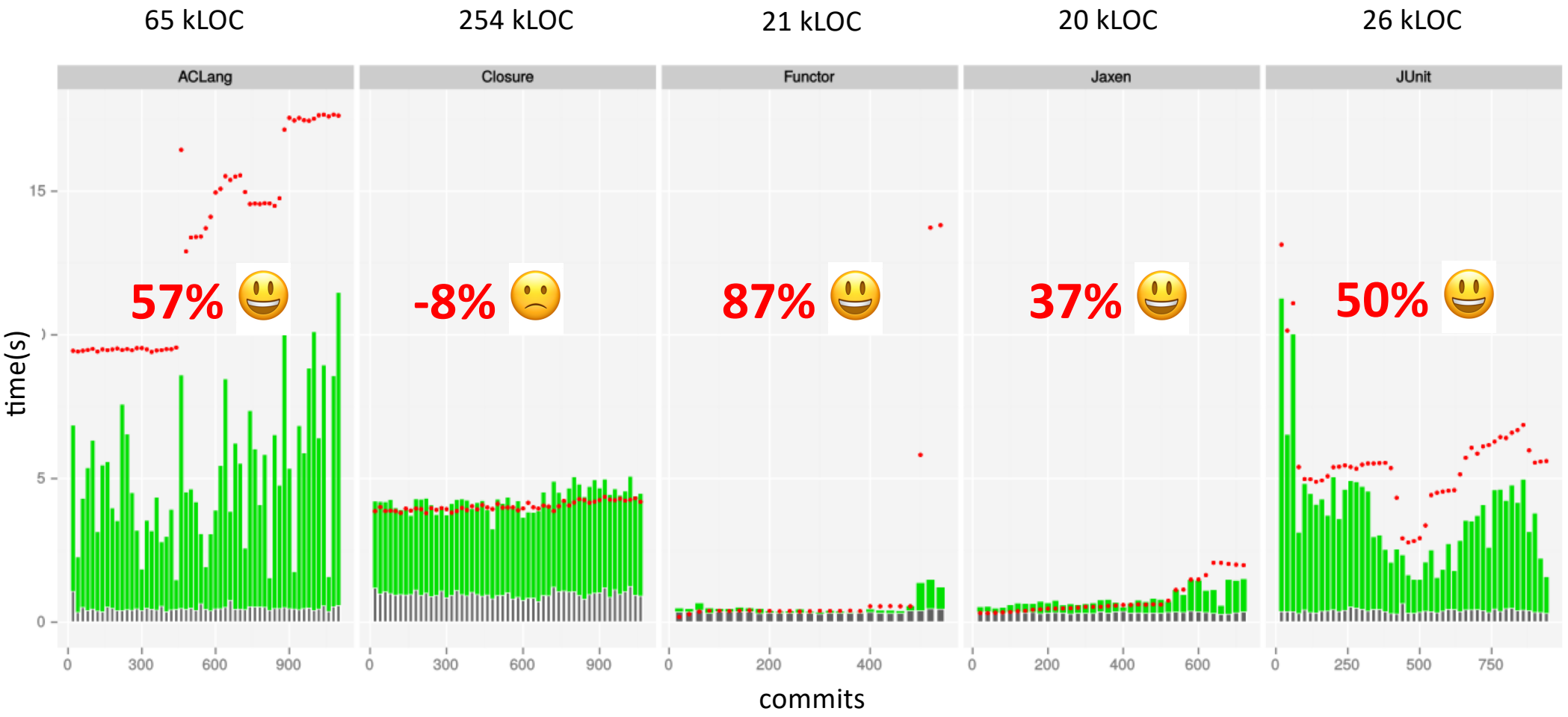
Based on the extensible Java compiler ExtendJ (JastAddJ).

The code is on Open Source: https://bitbucket.org/joqvist/autorts

Results on 5 Java projects



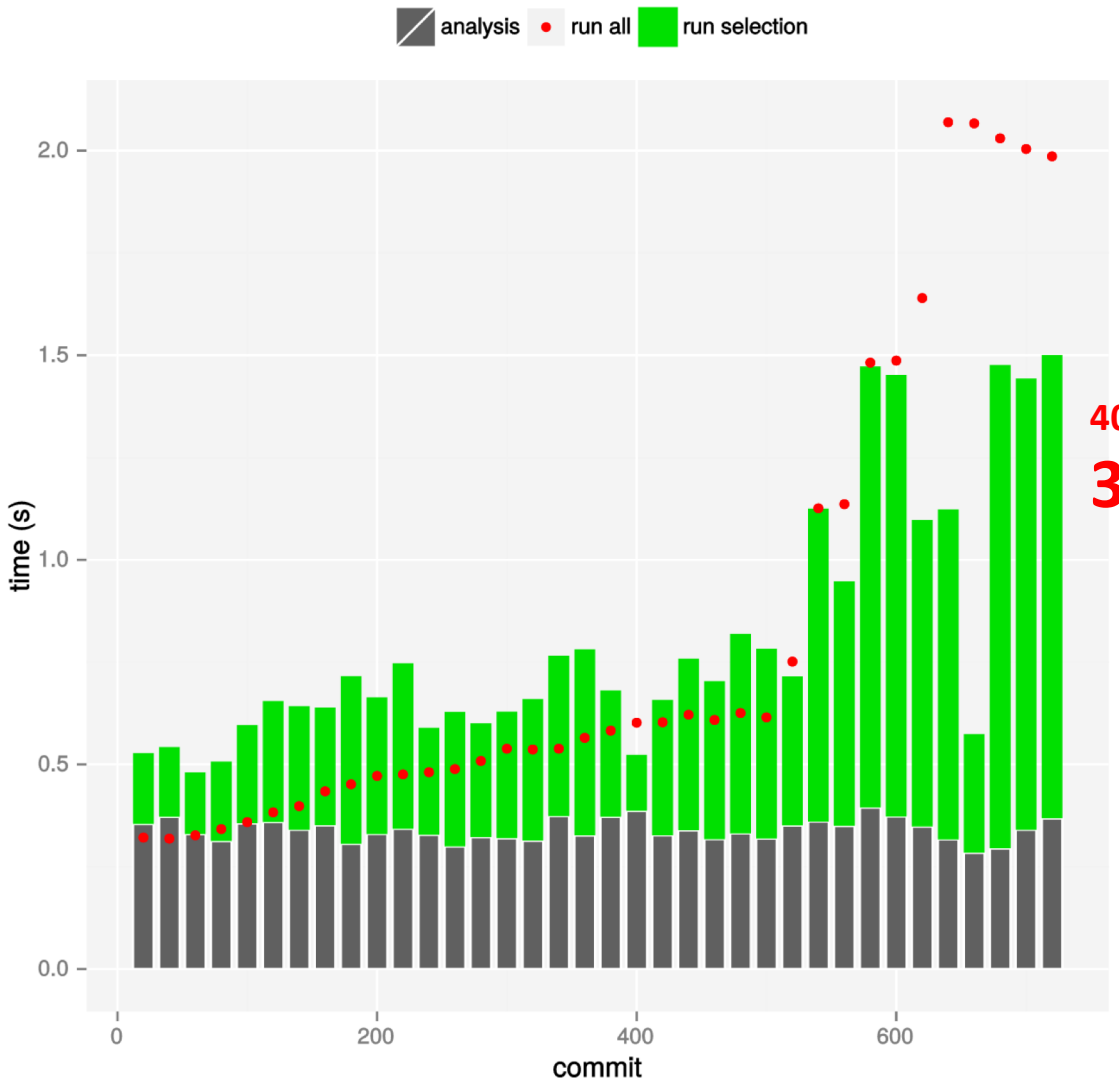
Results on 5 Java projects: RTS savings on last 40 commits



Jaxen

XML parsing library
20 kSLOC
~450 tests

- Short test run time
- Many changes trigger full re-runs
- Analysis fast but large % of test run time

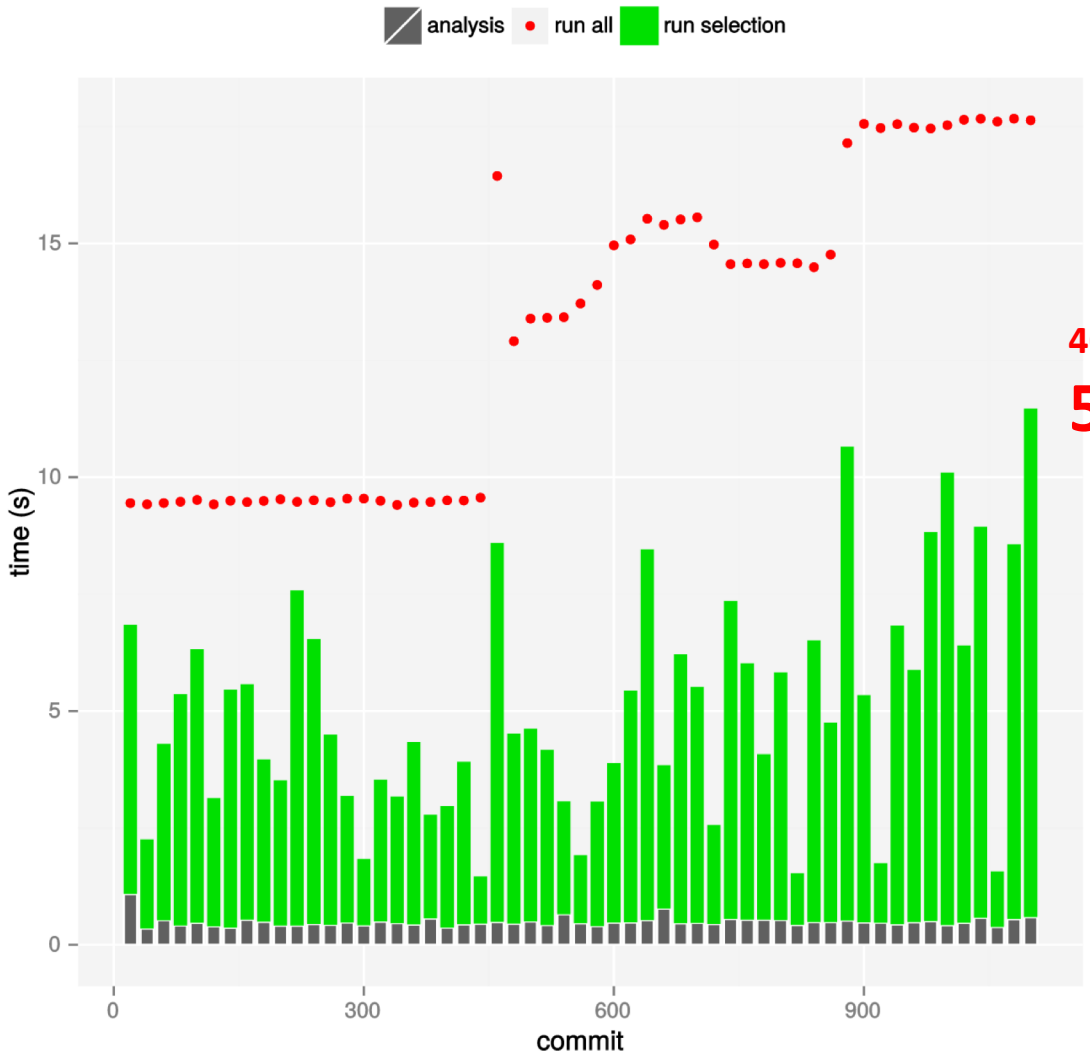


40 last commits
37% 😃

ACLang

Apache Commons Lang
65 kSLOC
~2500 tests

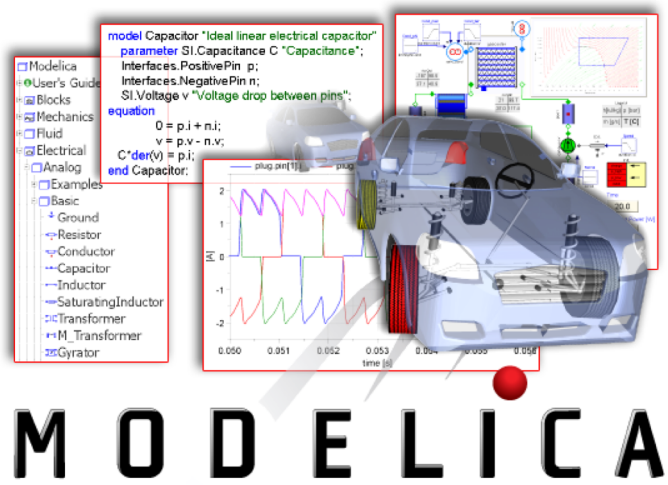
- Few changes trigger full re-runs
- Analysis small % of test run time



40 last commits
57% 😃

RTS for Modelica

- Two Master's Theses projects, in collaboration with Modelon
- Motivation: Testing Modelica Standard Library takes 2-3 hours
- Published at the American Modelica Conference, 2018



Modelica model

RTS Savings:

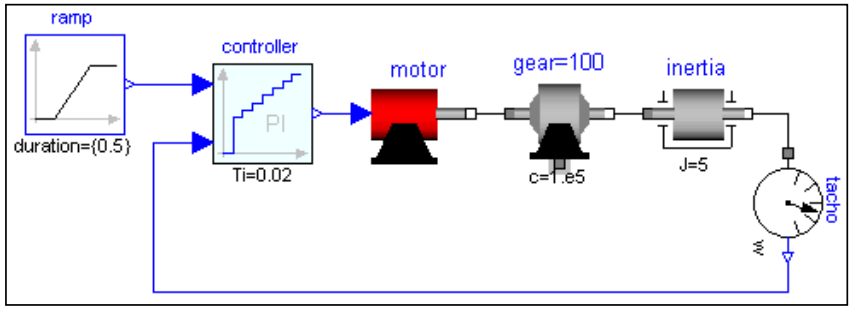
MSL (Modelica Standard Library): 96%

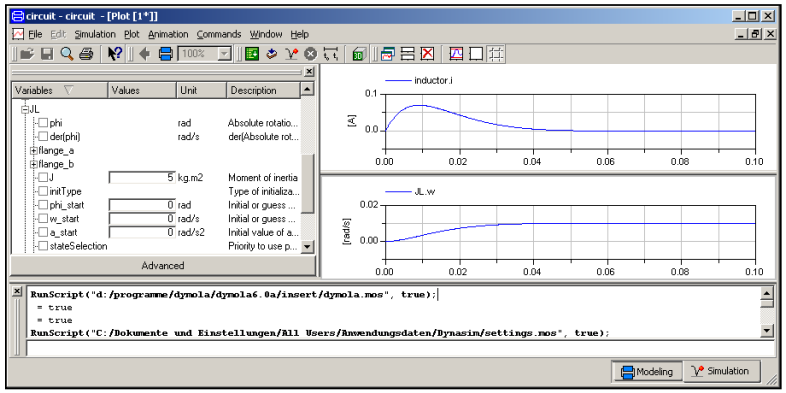
HXL (Heat Exchanger Library): 79%

```
model MotorDrive
  PI          controller;
  Ramp        ramp;
  Motor       motor;
  Gearbox     gear(ratio = 100);
  Inertia     inertia(J = 10);
  SpeedSensor tacho;
equation
  connect(controller.y    , motor.i_ref);
  connect(motor.flange    , gearbox.flange_a);
  connect(gearbox.flange_b, inertia.flange_a);
  connect(inertia.flange_b, tacho.flange);
  connect(tacho.w         , controller.u_m);
  connect(ramp.y          , controller.u_r);
end MotorDrive;
```

compile → simulate →

simulation result





Conclusions: Extraction-based RTS

Java

- Very fast analysis
- Pays off well on the average, for our studied Java projects
- Savings vary highly between projects, and during project lifetimes
- Works for program-based testing, not for input-based testing

Modelica

- Huge savings – simulations are much slower than unit tests – more to win

References

- Öqvist, Jesper, Görel Hedin, and Boris Magnusson. "Extraction-based regression test selection." Proceedings of the 13th International Conference on Principles and Practices of Programming on the Java Platform: Virtual Machines, Languages, and Tools. ACM, 2016.
- Fors, Niklas, Jon Sten, Markus Olsson, Filip Stenström. "A Safe Regression Test Selection Technique for Modelica". Proceedings of the American Modelica Conference, 2018