Racket Does Dijkstra

Systems Integrity Assurance via Weakest Precondition

Byron Davies, Ph.D.

OntoPilot LLC

OntoPilot.com

Agenda

- Goal: Fault-Free Code through Systems Integrity Assurance
- Thought Leaders
- Background: Dijkstra et al.
- Approach: Weakest Precondition
- Examples
- Racket Software Demo

Goal: Fault-Free Software Maintenance

- Bugs have been present since the invention of the stored-program computer

Goal: Fault-Free Software Maintenance

- Bugs have been present since the invention of the stored-program computer
- Some approaches: strong typing, testing, model checking

Goal: Fault-Free Software Maintenance

- Bugs have been present since the invention of the stored-program computer
- Some approaches: strong typing, testing, model checking
- All of these have to do with developing new code

Goal: Fault-Free Software Maintenance

- Bugs have been present since the invention of the stored-program computer
- Some approaches: strong typing, testing, model checking
- All of these have to do with developing new code
- What if there were a method that could find bugs in legacy code?

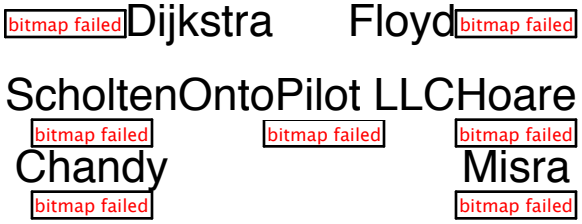
Goal: Fault-Free Software Maintenance

- Bugs have been present since the invention of the stored-program computer
- Some approaches: strong typing, testing, model checking
- All of these have to do with developing new code
- What if there were a method that could find bugs in legacy code?
- What if a variation of this method could also help you figure out what the legacy code does?

Goal: Fault-Free Software Maintenance

- Bugs have been present since the invention of the stored-program computer
- Some approaches: strong typing, testing, model checking
- All of these have to do with developing new code
- What if there were a method that could find bugs in legacy code?
- What if a variation of this method could also help you figure out what the legacy code does?
- We call this method SIA, for Systems Integrity Assurance

SIA Thought Leaders



- Floyd applied logic to programming
- Hoare invented notation for pre-condition and post-condition
- Dijkstra advocated for the predicate transformer, weakest precondition
- Scholten help him formalize the mathematics
- Chandy and Misra extended the approach to non-deterministic/parallel programs

Weakest Precondition for sequential code

- Use predicate Q to specify a postcondition on the "state space"
- Use these rules to determine the weakest precondition that will imply Q
- (wp (no-op) Q) = Q
- (wp (set! x e) Q) = (subst e x Q)
- (wp (begin e1 e2) Q) = (wp e1 (wp e2 Q))
- (wp (if B e1 e2) Q) = (or (and B e1) (and (not B) e2))
- Etc.

Summary of WP

The road to hell is paved with ...

Summary of WP

The road to hell is paved with ...

... bad assumptions

Summary of WP

The road to hell is paved with ...

... bad assumptions

You need to CYA -- Check Your Assumptions

Special Case: Slicing

- Goal: figure out how the outputs depend on the inputs
- Question posed: under what conditions will this code produce these abstract values?
- Use a special post-condition of the form "var1=a and var2=b ..."

General Version: Dijkstra Guarded Commands

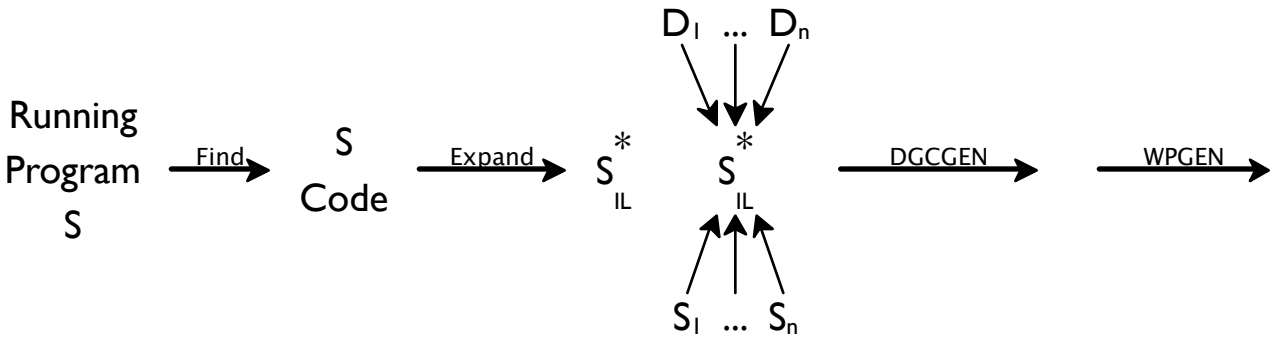
- The nondeterministic case:

```
(dgc guard-exp exp ...)

(dgc-if (dgc guard-exp exp ...) ...)

(dgc-do (dgc guard-exp exp ...) ...)

(s-assign (x ...) (exp ...))
```

Assist from Micron Automata Processor?

- http://MicronAutomata.com
- Initial SIA project: WP + AP => fault-free code, fast
- AP: Array of 48K state transition elements per chip, 1.5M/board
- Capable of recognizing many thousands of complex patterns in real-time in near-gigibit data stream
- Up for a challenge?

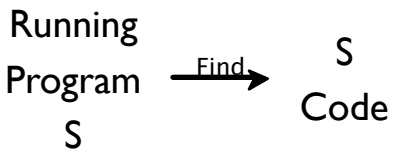
Tool Development
for
System Integrity Analysis



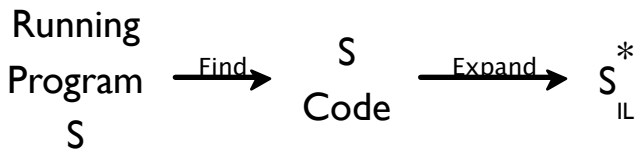
Tool Development
for
System Integrity Analysis

Running
Program
S

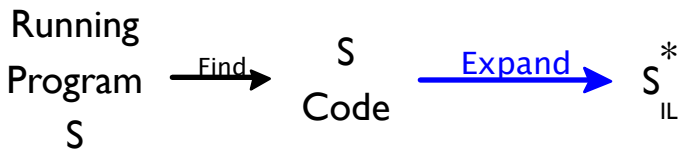
Tool Development
for
System Integrity Analysis



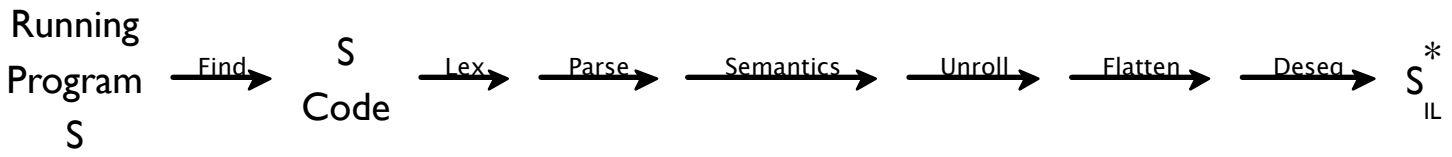
Tool Development
for
System Integrity Analysis



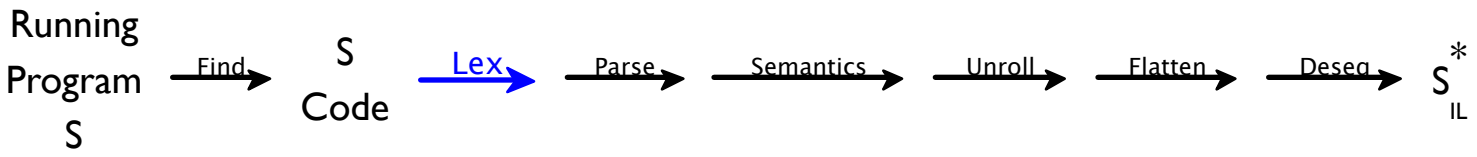
Tool Development
for
System Integrity Analysis



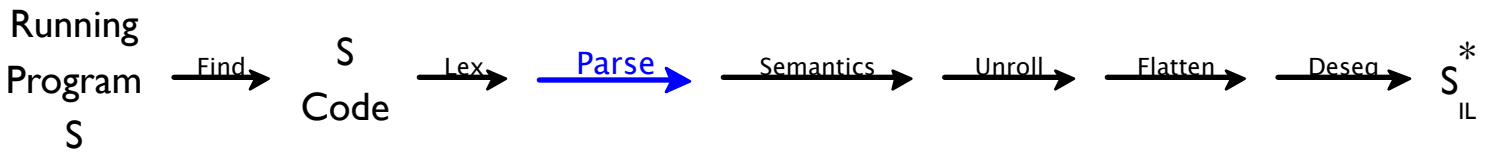
Tool Development
for
System Integrity Analysis



Tool Development
for
System Integrity Analysis

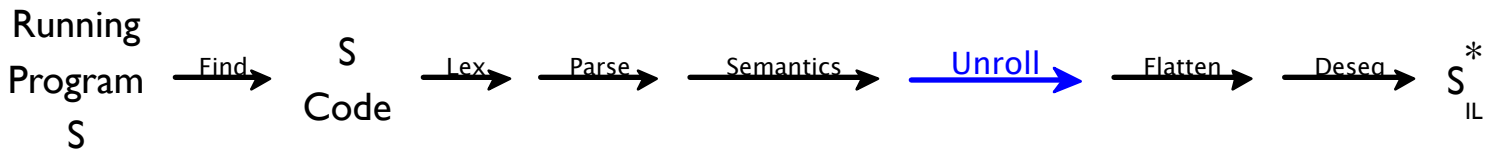


Tool Development
for
System Integrity Analysis

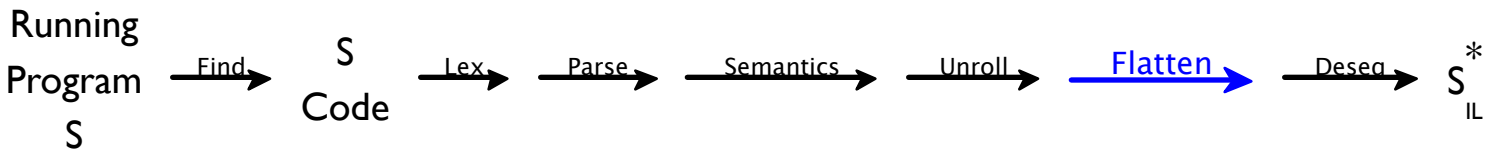


Tool Development
for
System Integrity Analysis

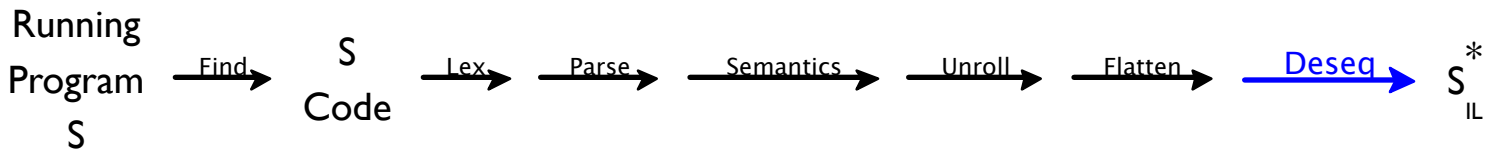
Tool Development
for
System Integrity Analysis



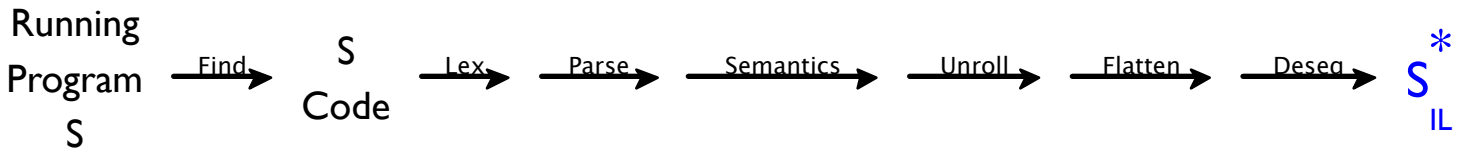
Tool Development
for
System Integrity Analysis



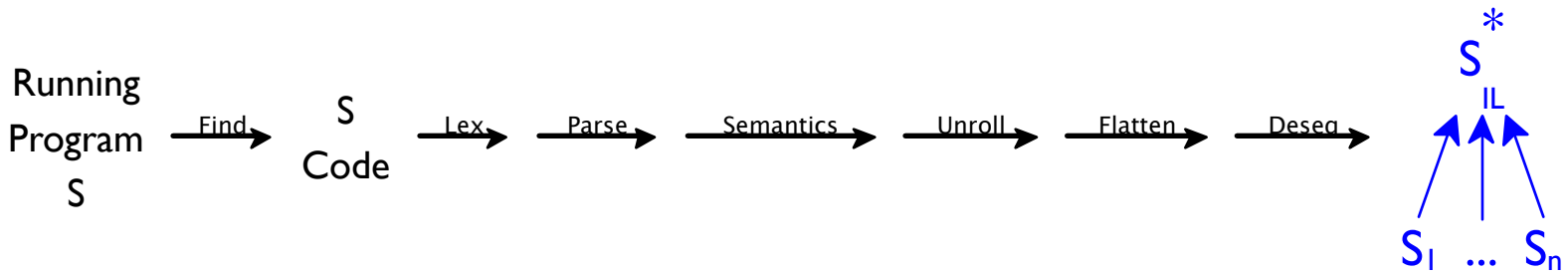
Tool Development
for
System Integrity Analysis



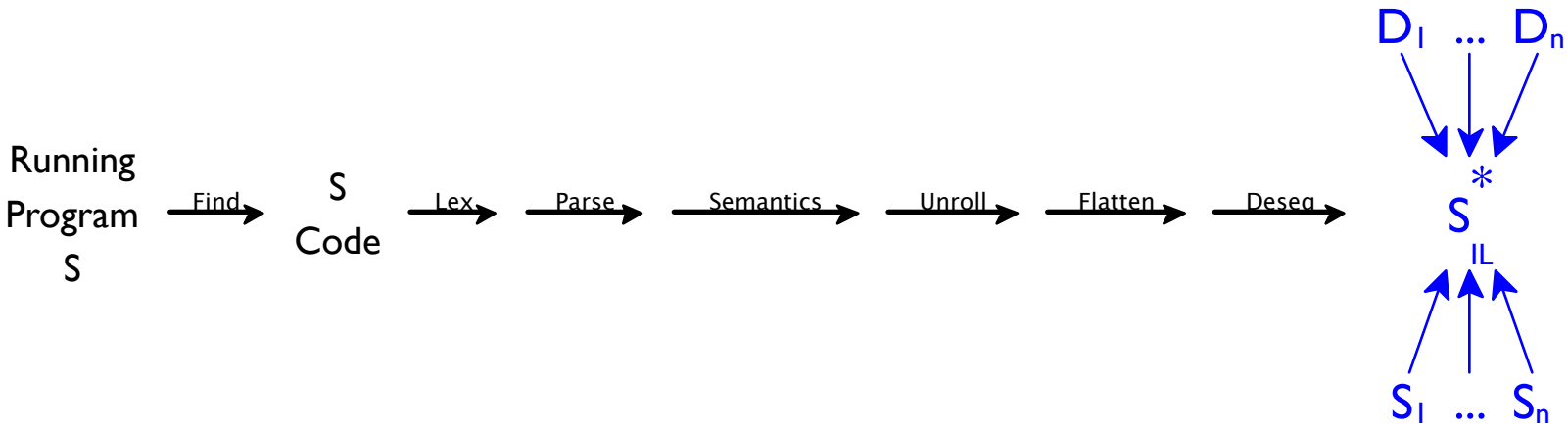
Tool Development
for
System Integrity Analysis



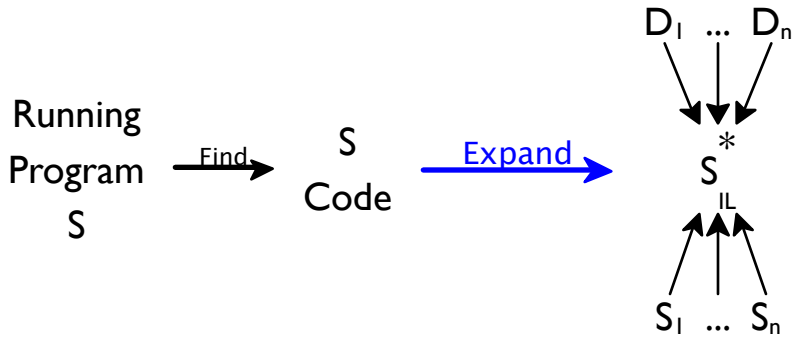
Tool Development
for
System Integrity Analysis



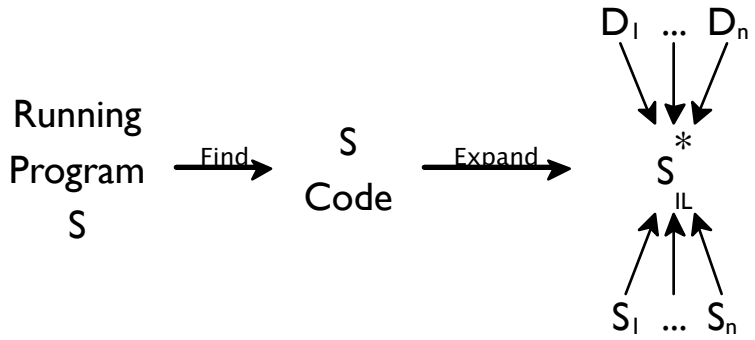
Tool Development
for
System Integrity Analysis



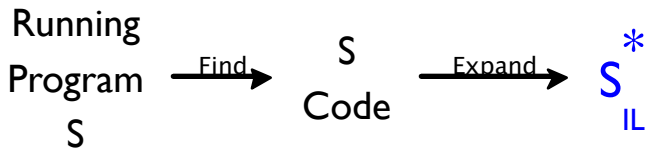
Tool Development
for
System Integrity Analysis



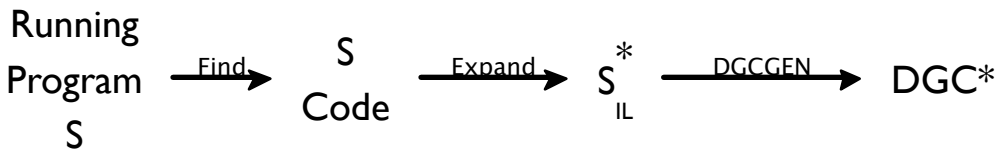
Tool Development
for
System Integrity Analysis



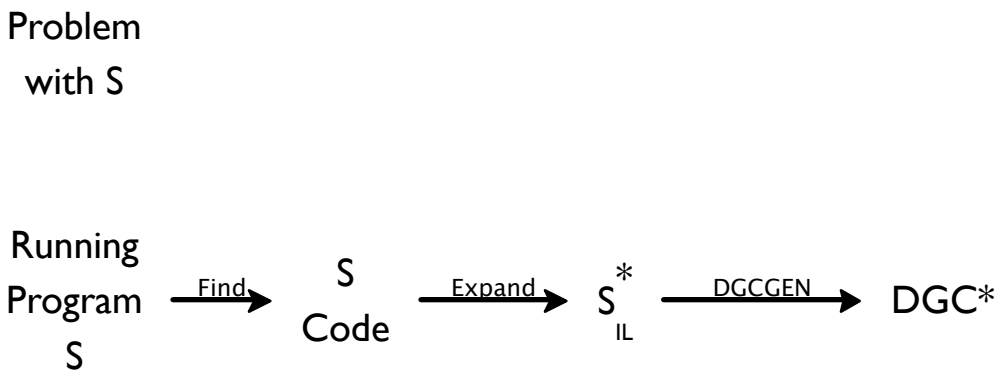
Tool Development
for
System Integrity Analysis



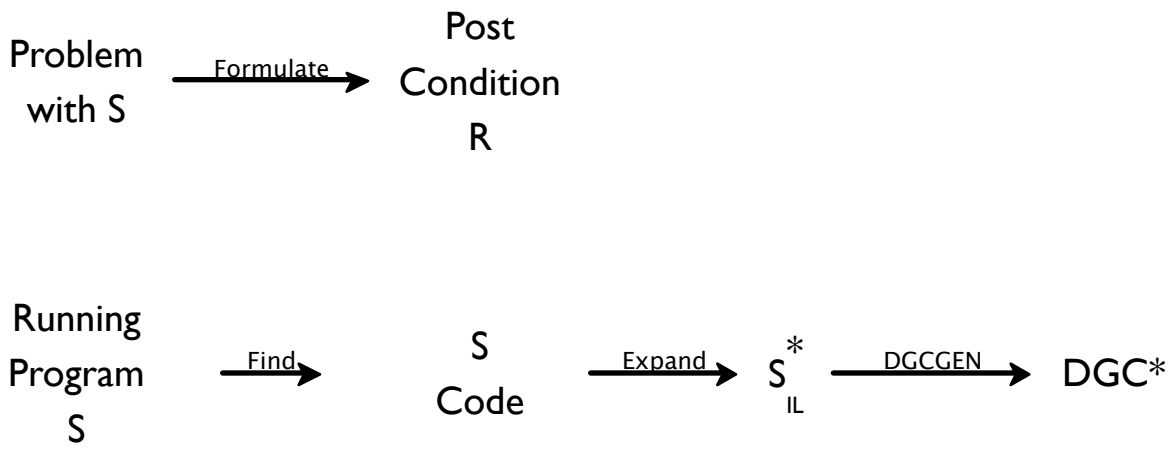
Tool Development
for
System Integrity Analysis



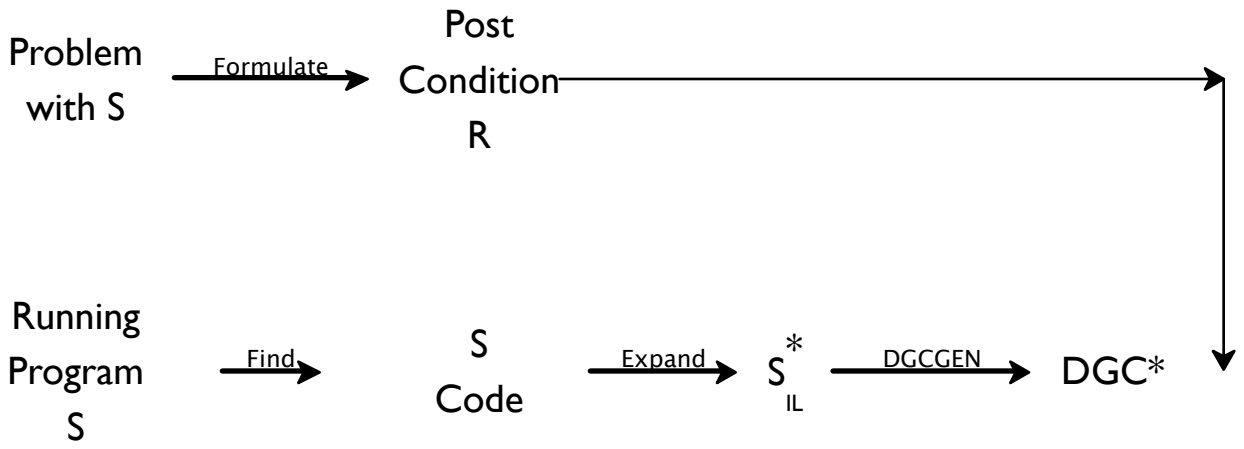
Tool Development
for
System Integrity Analysis



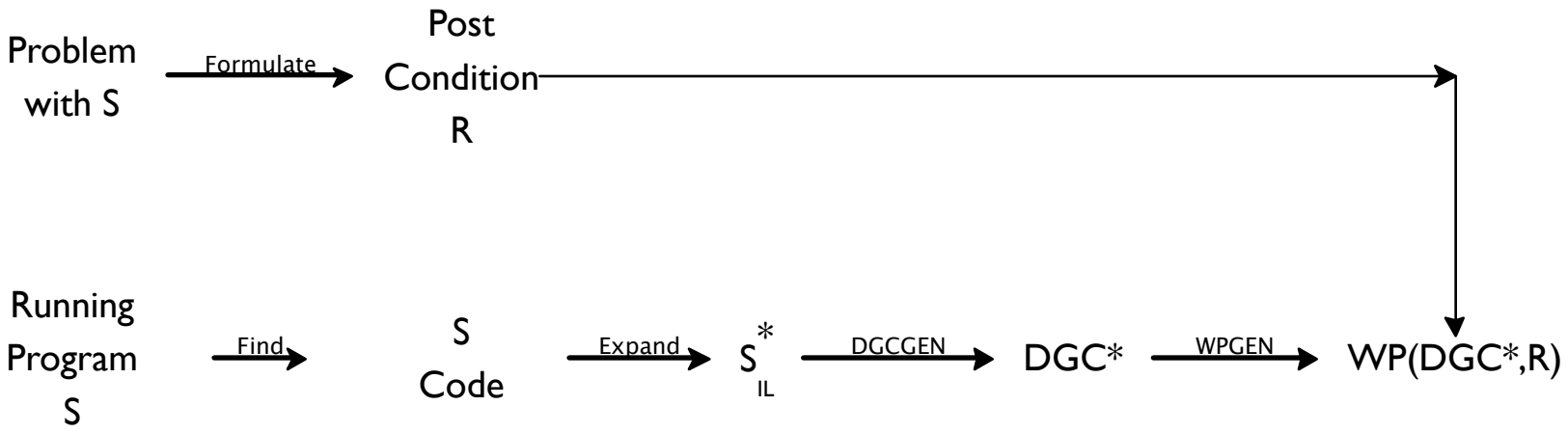
Tool Development
for
System Integrity Analysis



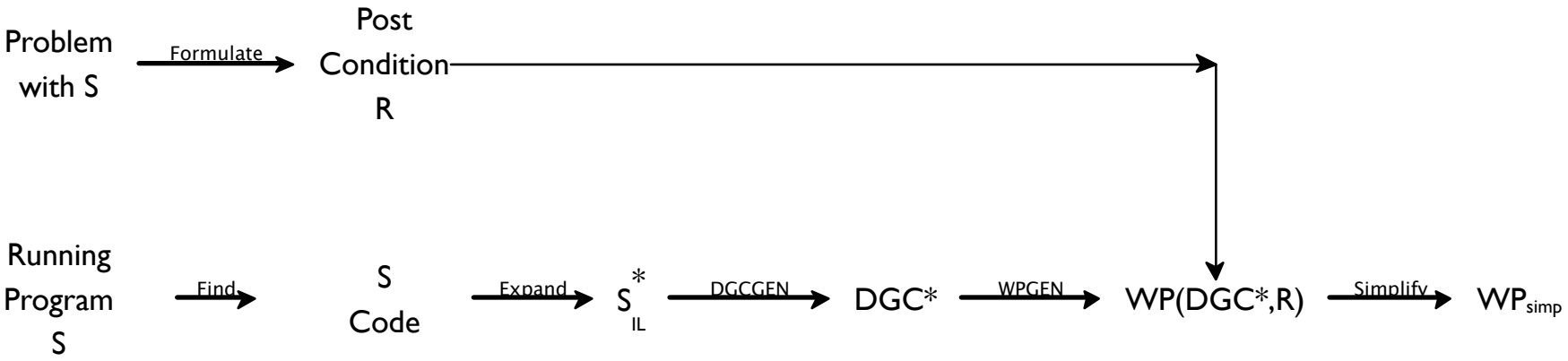
Tool Development
for
System Integrity Analysis



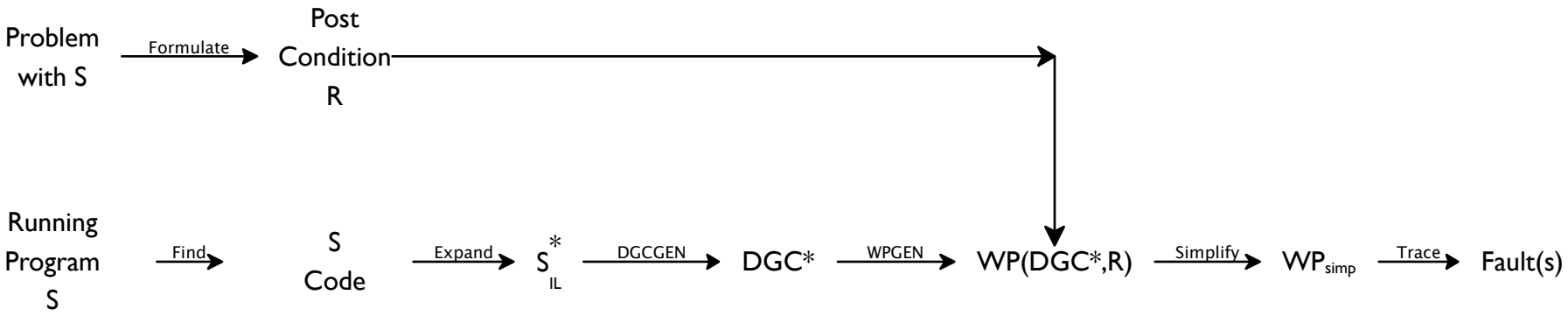
Tool Development
for
System Integrity Analysis



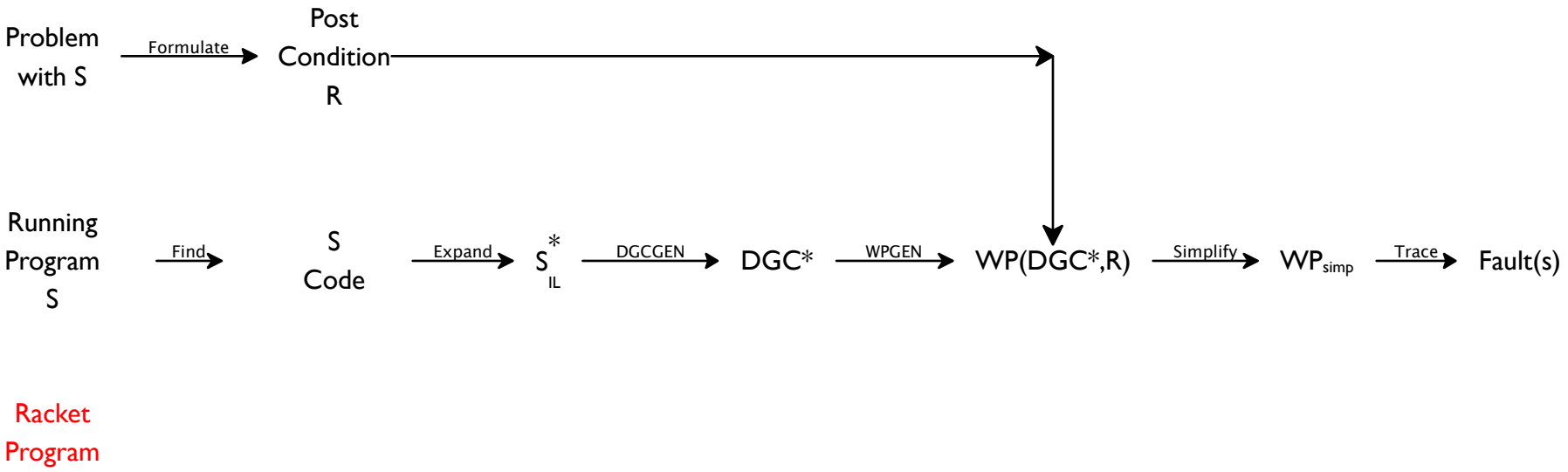
Tool Development
for
System Integrity Analysis



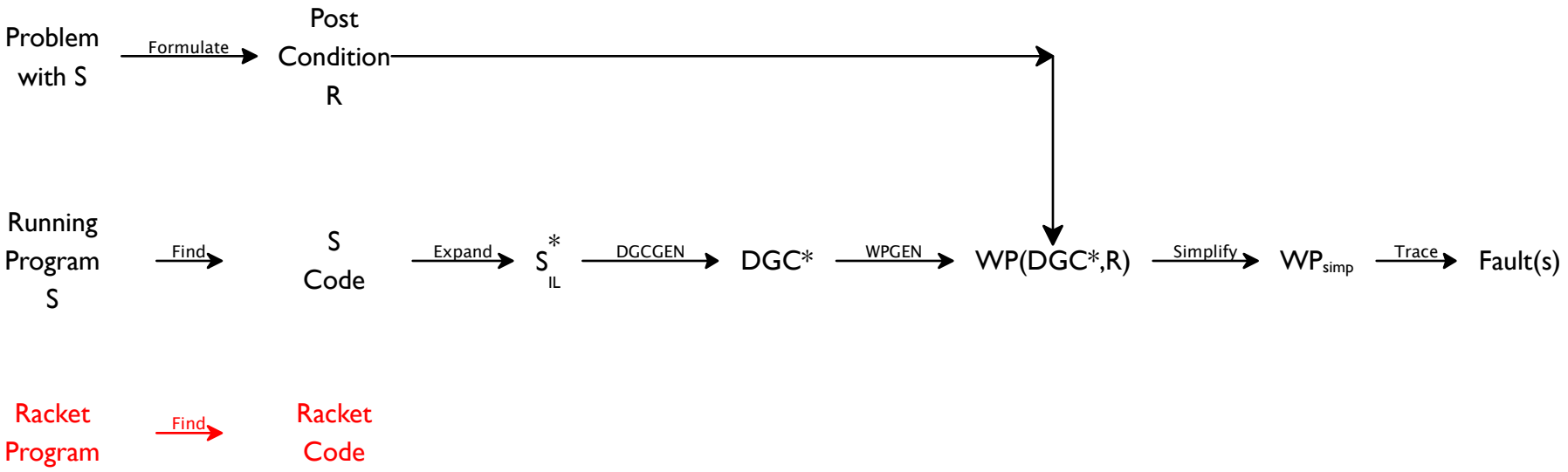
Tool Development
for
System Integrity Analysis



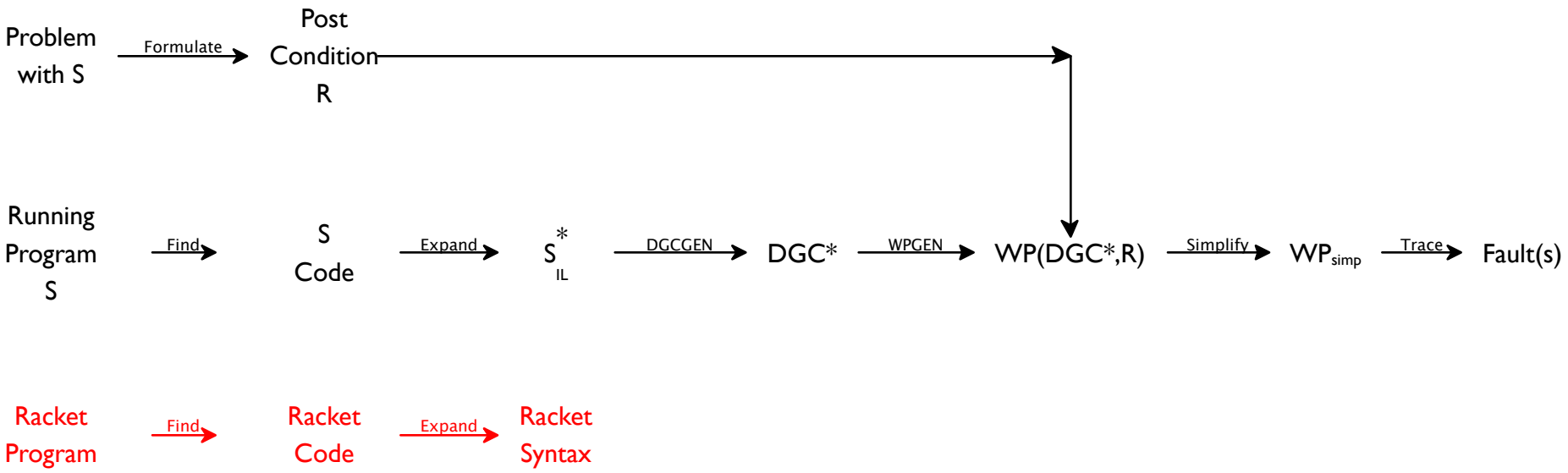
Tool Development
for
System Integrity Analysis



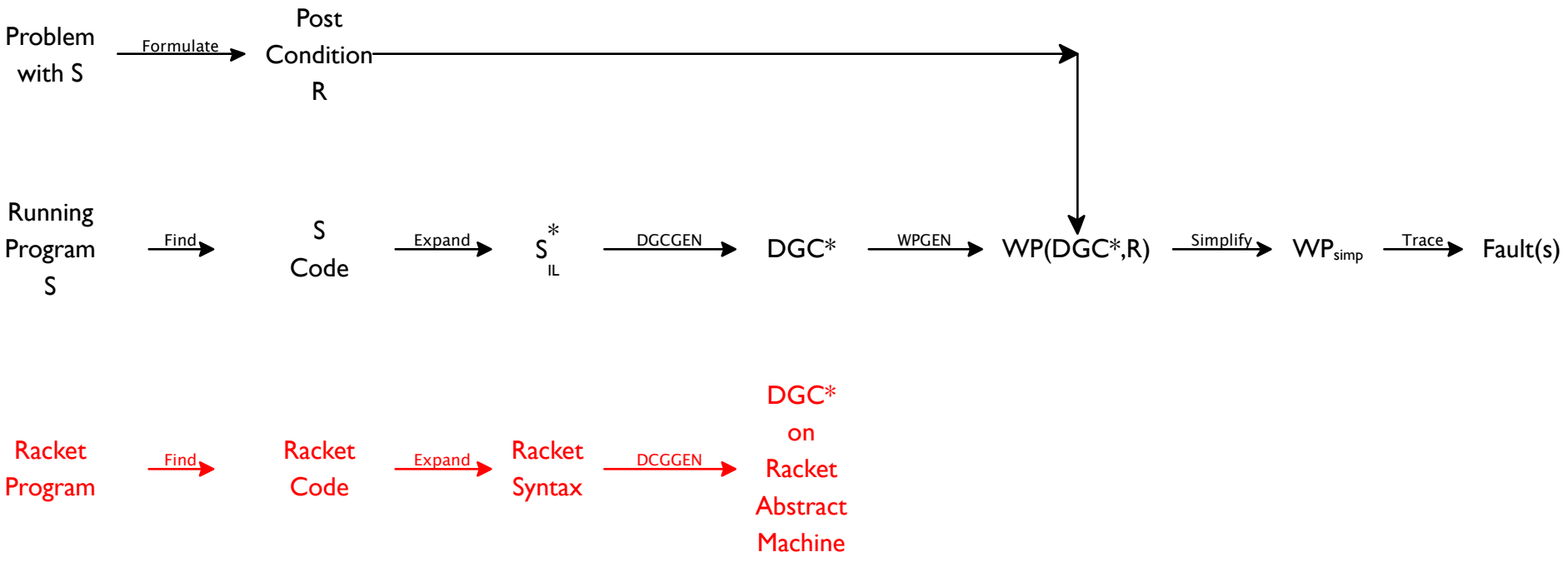
Tool Development
for
System Integrity Analysis



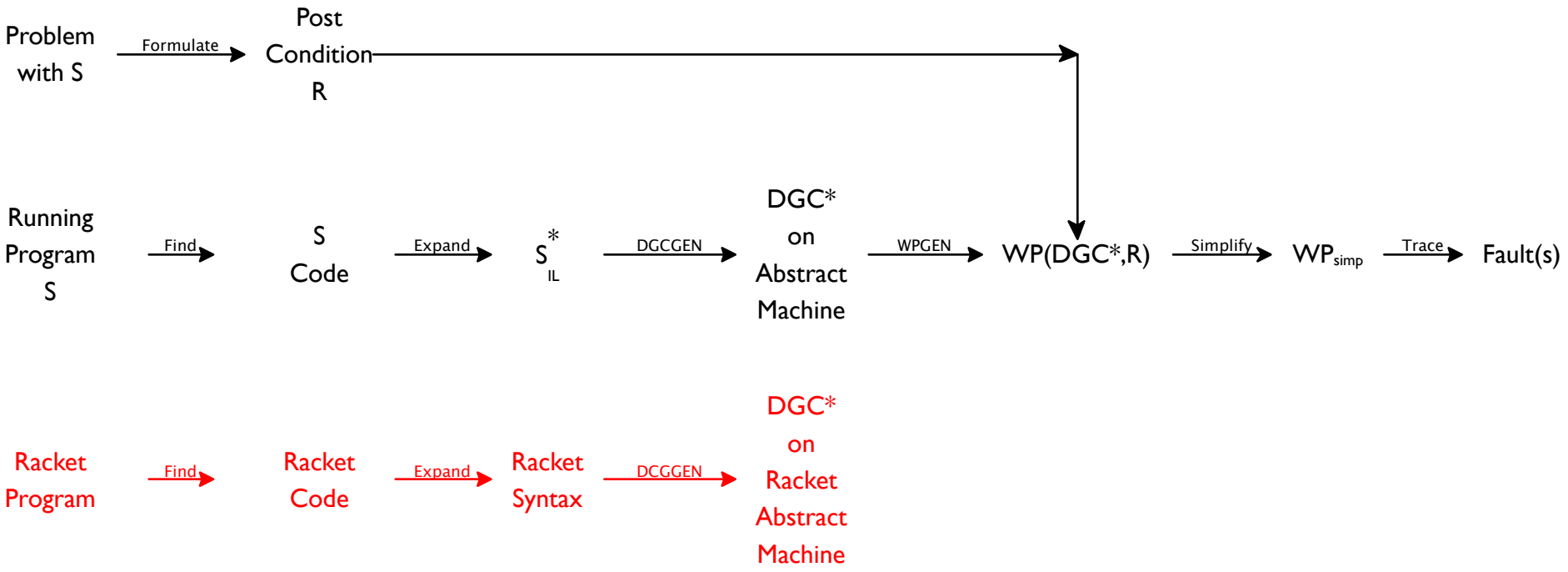
Tool Development
for
System Integrity Analysis



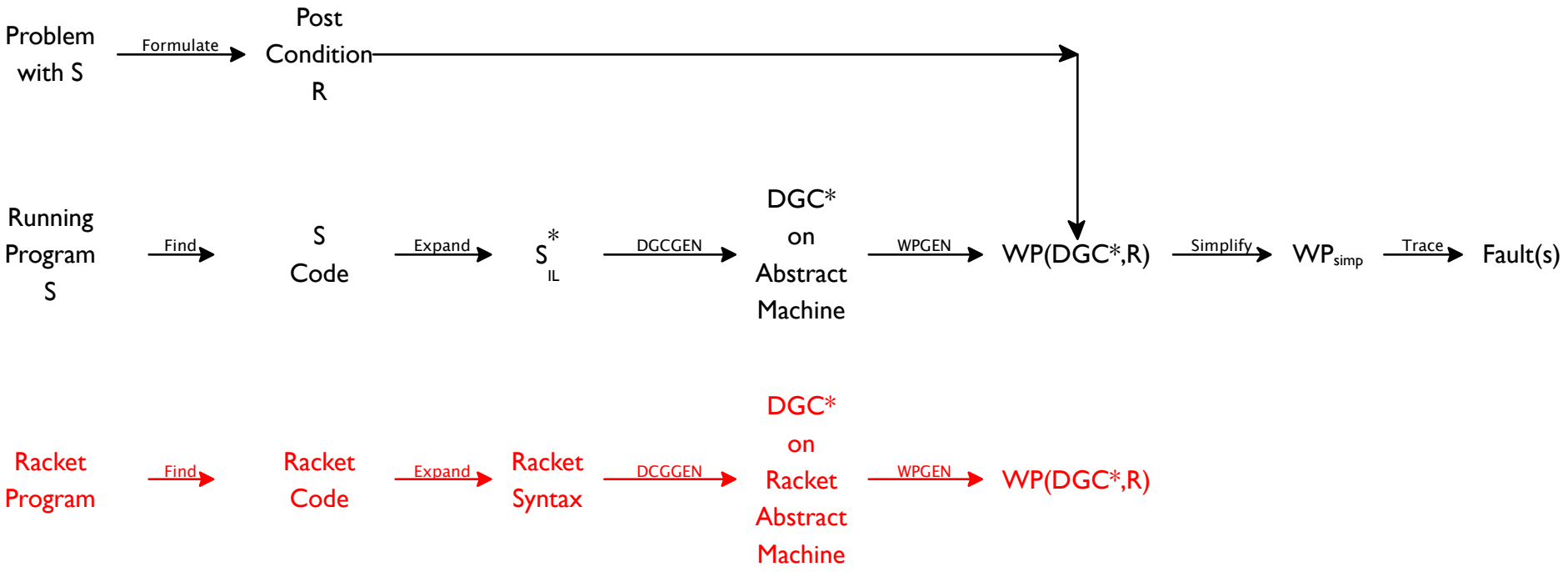
Tool Development
for
System Integrity Analysis



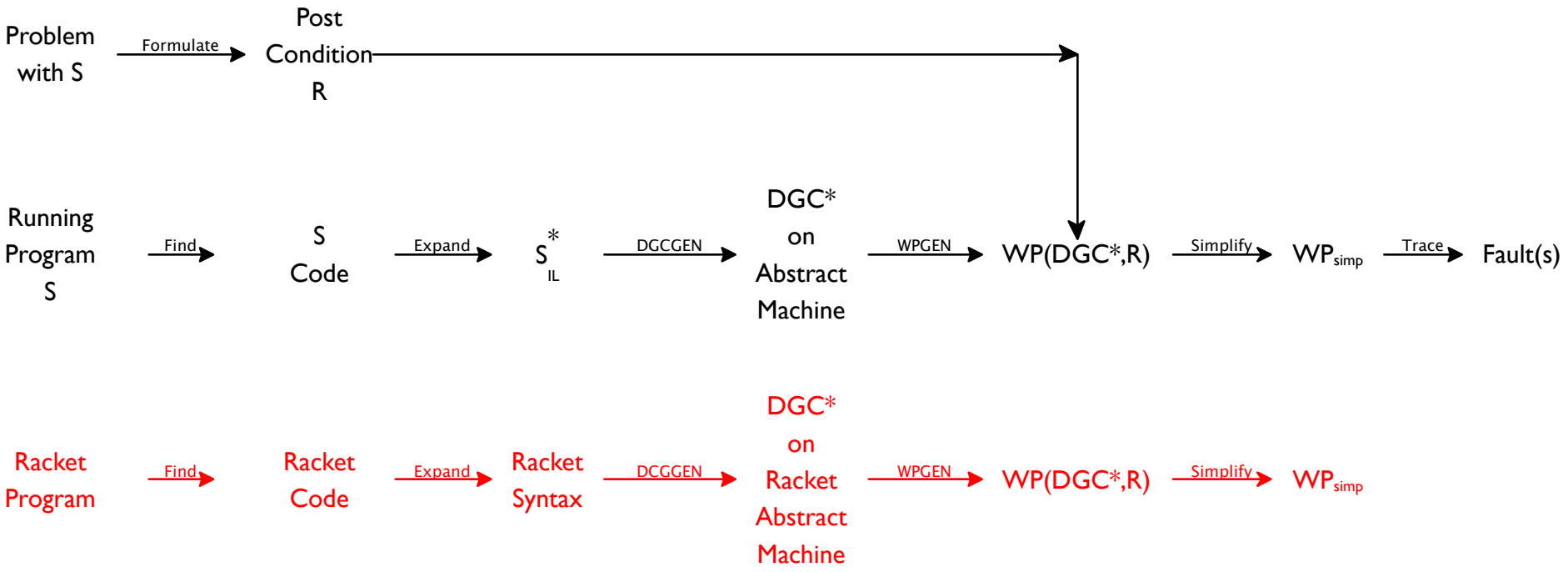
Tool Development
for
System Integrity Analysis



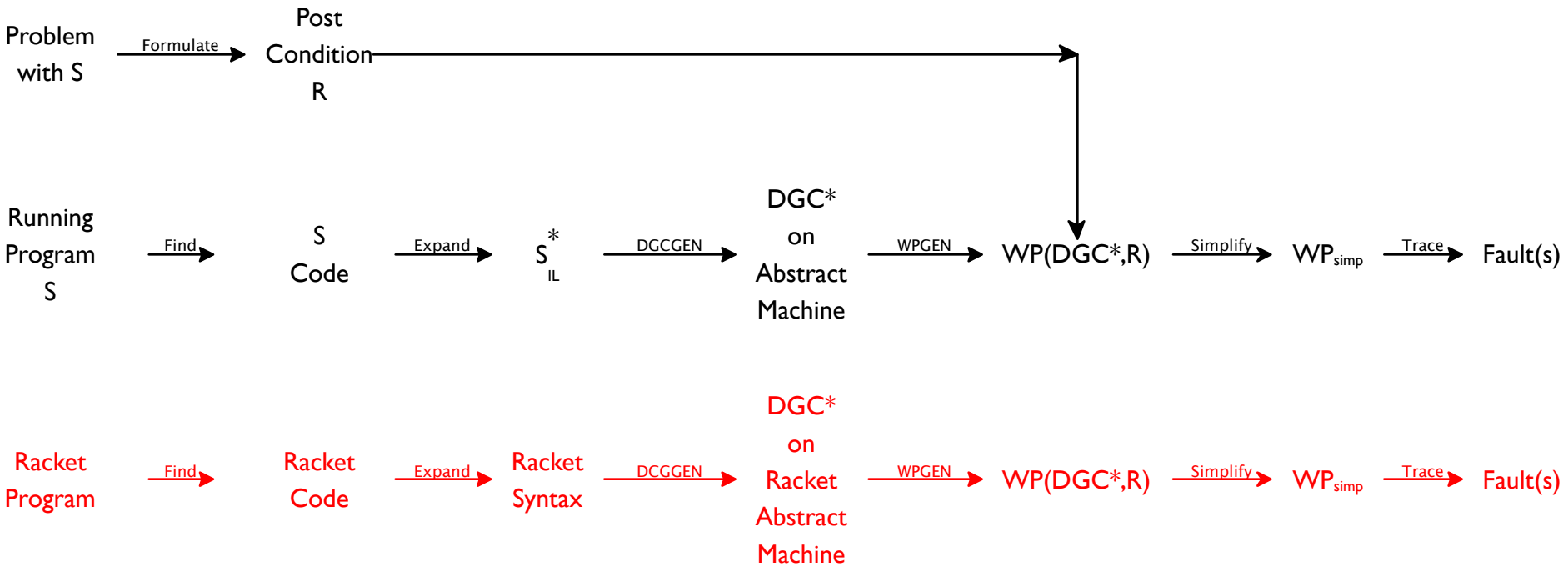
Tool Development
for
System Integrity Analysis



Tool Development
for
System Integrity Analysis



Tool Development
for
System Integrity Analysis



Demo

Collaboration?

- Current OntoPilot business model: license our IP to start-ups in different vertical markets

Collaboration?

- Current OntoPilot business model: license our IP to start-ups in different vertical markets
- One so far, in enterprise Java software

Collaboration?

- Current OntoPilot business model: license our IP to start-ups in different vertical markets
- One so far, in enterprise Java software
- Issue: ideas are mature (we think), but implementing it seems too challenging

Collaboration?

- Current OntoPilot business model: license our IP to start-ups in different vertical markets
- One so far, in enterprise Java software
- Issue: ideas are mature (we think), but implementing it seems too challenging
- We really want to get the capability in use

Collaboration?

- Current OntoPilot business model: license our IP to start-ups in different vertical markets
- One so far, in enterprise Java software
- Issue: ideas are mature (we think), but implementing it seems too challenging
- We really want to get the capability in use
- We would love to share with the research community to take it to the next level