UMR IRISA

Reconciling Compiler Optimizations and WCET Estimation Using Iterative Compilation

Mickaël Dardaillon, **Stefanos Skalistis**, Isabelle Puaut, Steven Derrien

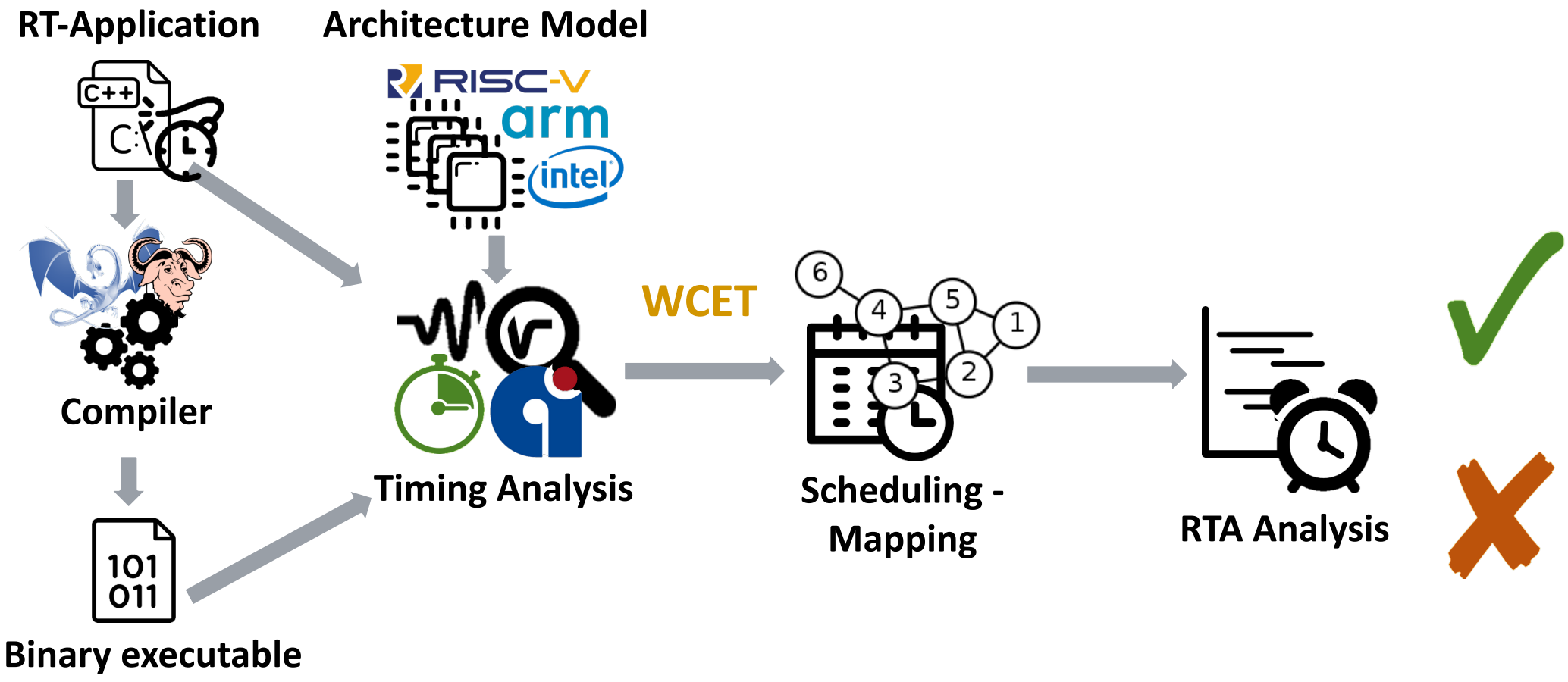
RTSS'19, Hong Kong

Dec. 2019

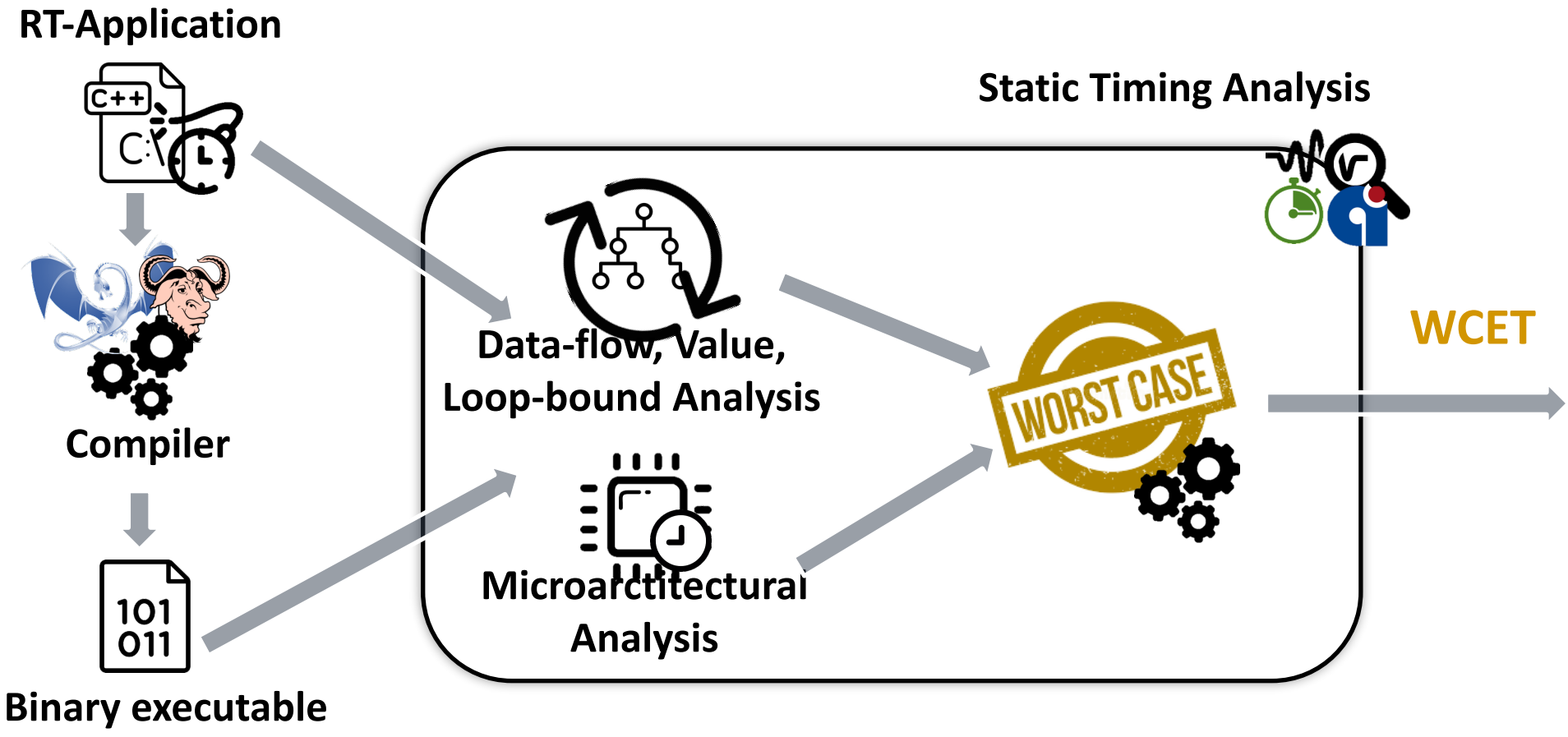
Institut de Recherche en Informatique et Systèmes Aléatoires

CentraleSupélec · cnrs · ENS rennes · Inria · IMT Atlantique Bretagne-Pays de la Loire École Mines-Télécom · INSA RENNES · Université Bretagne Sud ubs · UNIVERSITÉ DE RENNES 1

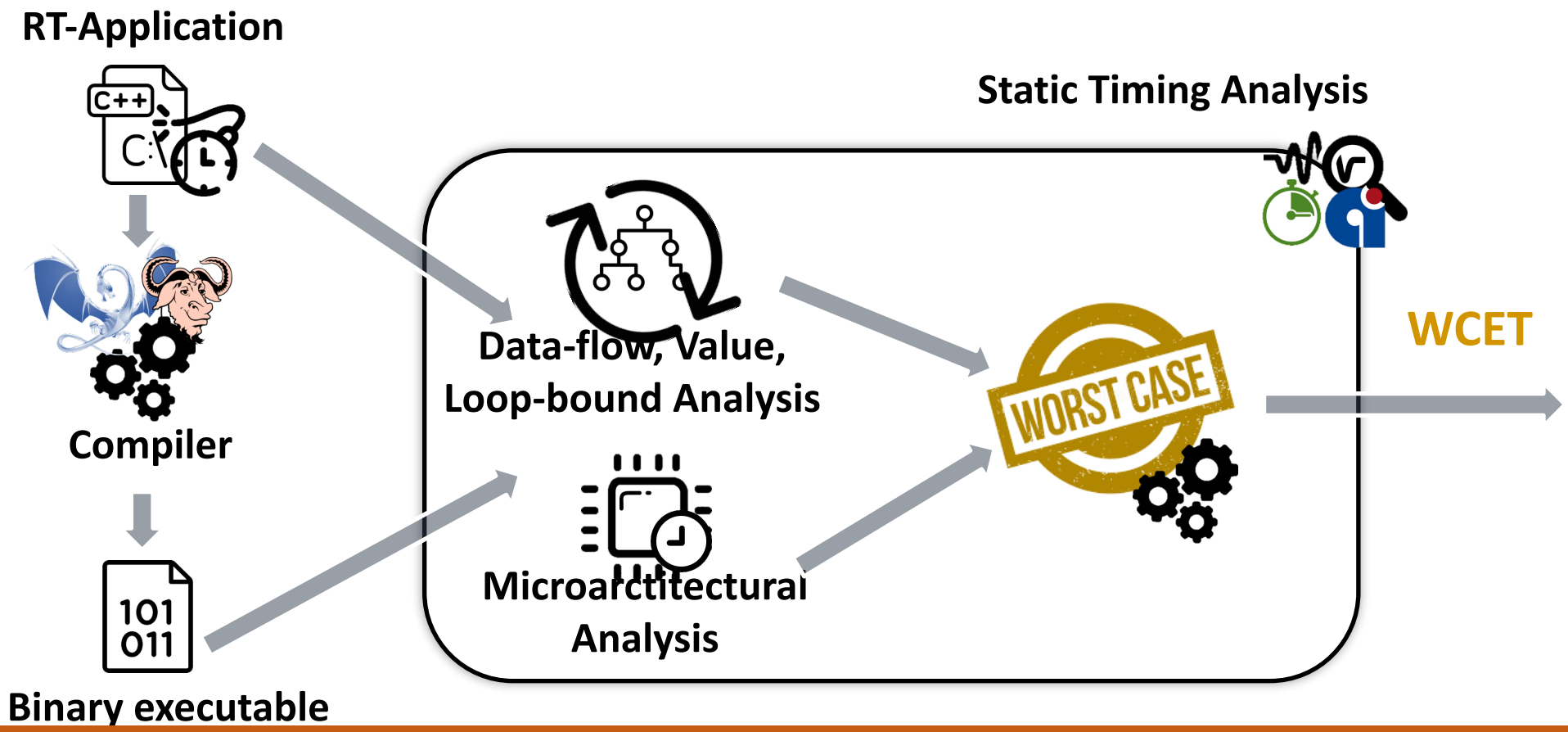
Typical Real-time workflow



Static WCET analysis



Static WCET analysis



… highly depends on the compiler as well!

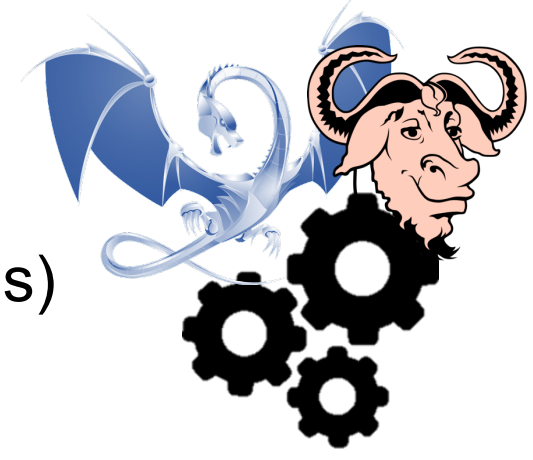
Compiler optimisations

- **WCET-aware compilers**
  - Limited number of optimisations
  - Limited combinations of optimisations
  - Suitable for real-time

jcert

WCC

- **Mainstream compilers:**
  - High number of optimisations
  - Allow any optimisation sequence (flags/passes)
  - No knowledge of optimisations impact
  - Can break static WCET analyzability

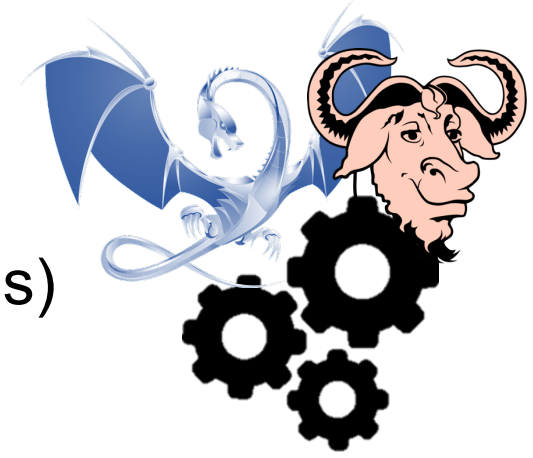


Compiler optimisations

- **WCET-aware compilers**
  - Limited number of optimisations
  - Limited combinations of optimisations
  - Suitable for real-time

- **Mainstream compilers:**
  - High number of optimisations
  - Allow any optimisation sequence (flags/passes)
  - No knowledge of optimisations impact
  - Can break static WCET analyzability

WCC

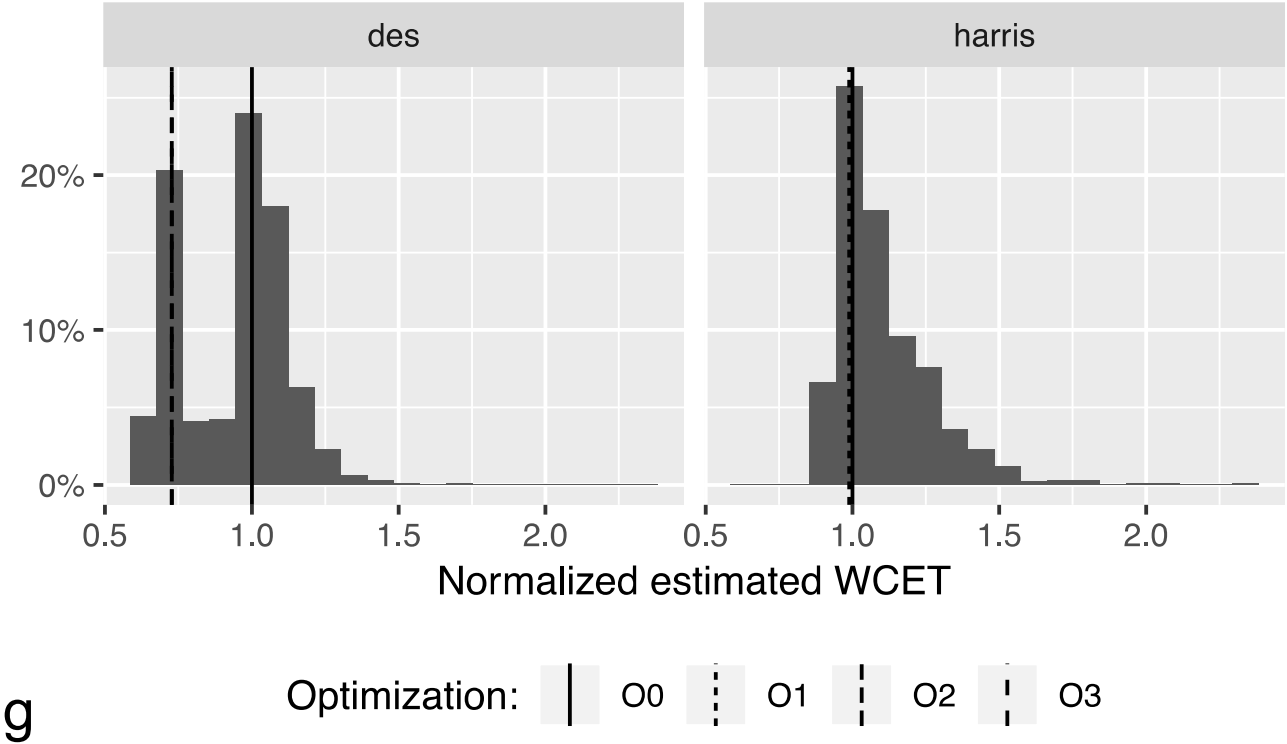


Can mainstream compilers be utilised?

WCET impact of compiler optimisations

1000 random sequences of optimisation passes:

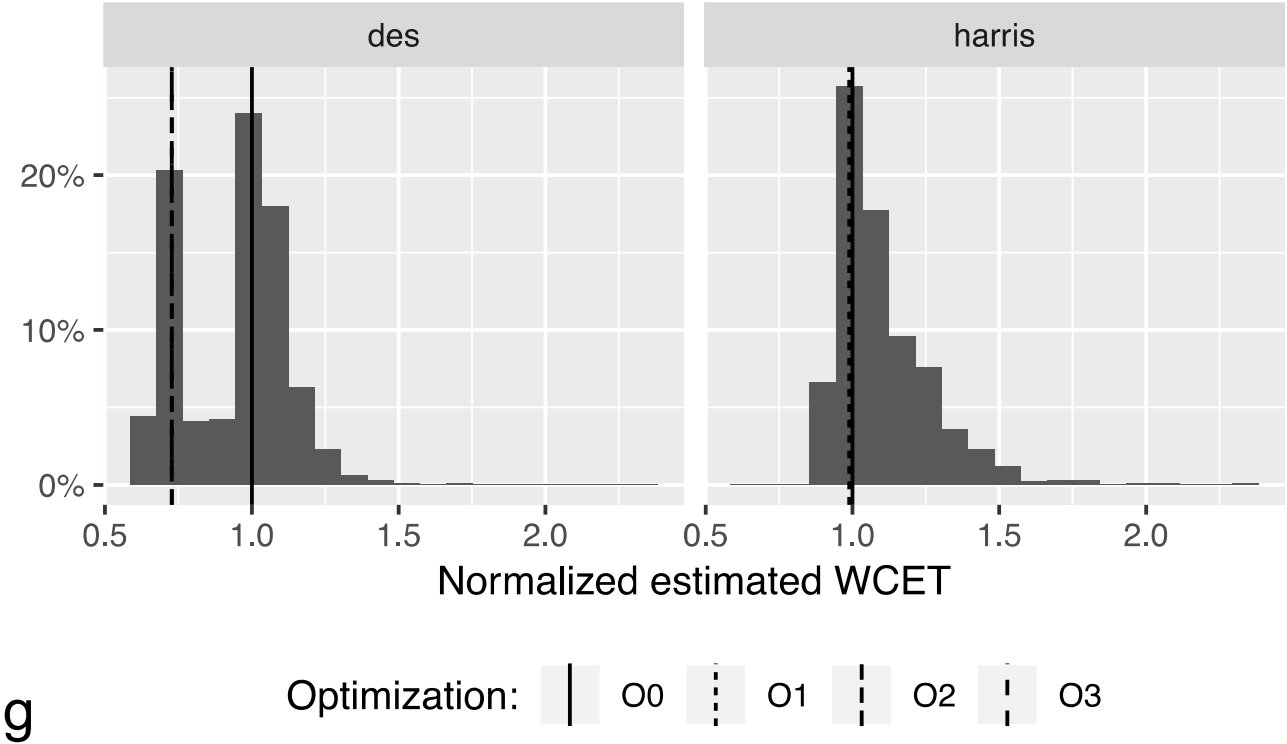
- *Compiler:* LLVM for Leon3
- *Static WCET*: AbsInt aiT
- *Des:* embedded application (Mälardalen Benchmarks)
- *Harris:* image processing application (PolyMage)



WCET impact of compiler optimisations

1000 random sequences of optimisation passes:

- *Compiler:* LLVM for Leon3
- *Static WCET*: AbsInt aiT
- *Des:* embedded application (Mälardalen Benchmarks)
- *Harris:* image processing application (PolyMage)

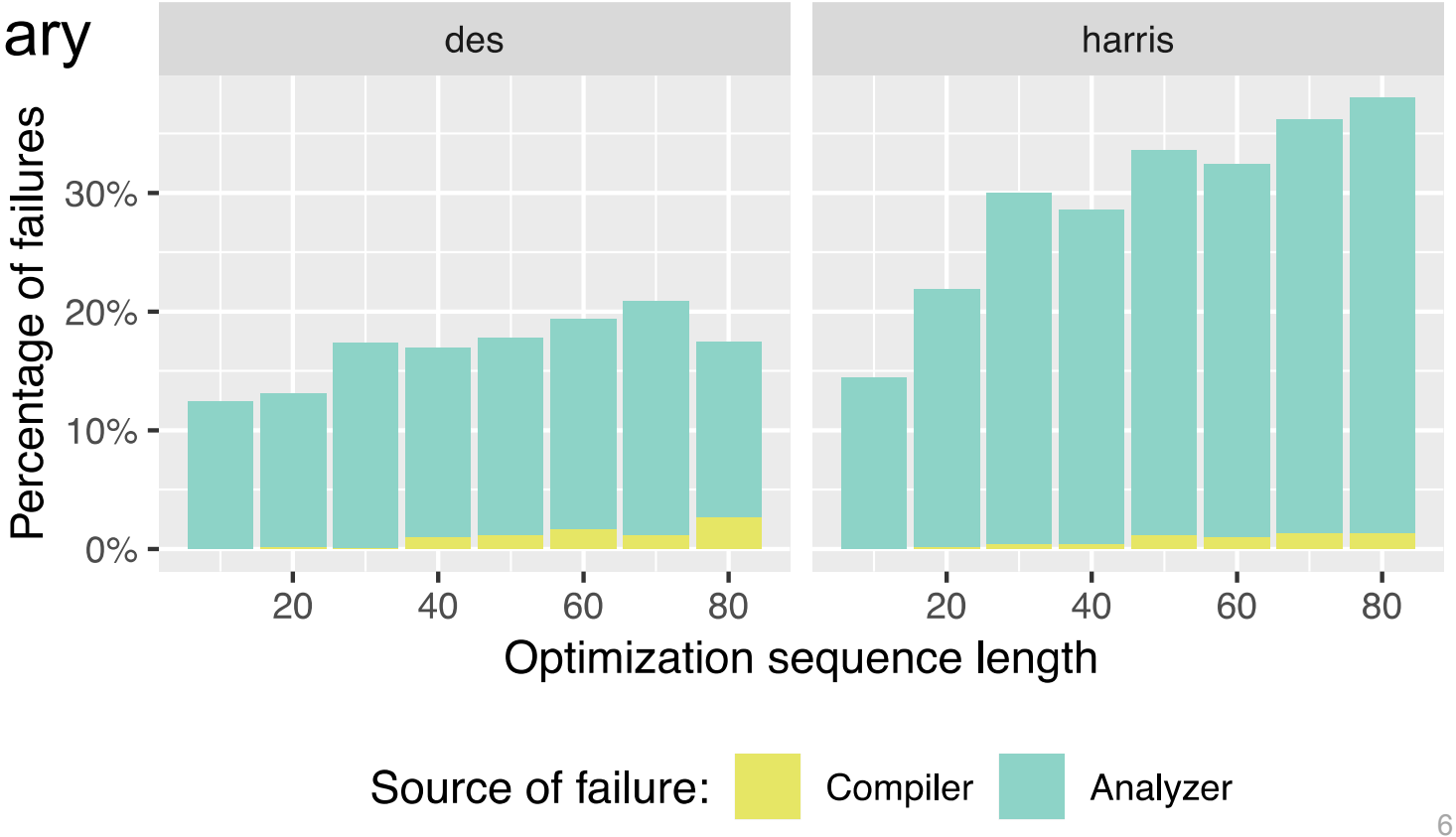


Optimisations can worsen WCET

Impact on analyzability

Different Optimisations:

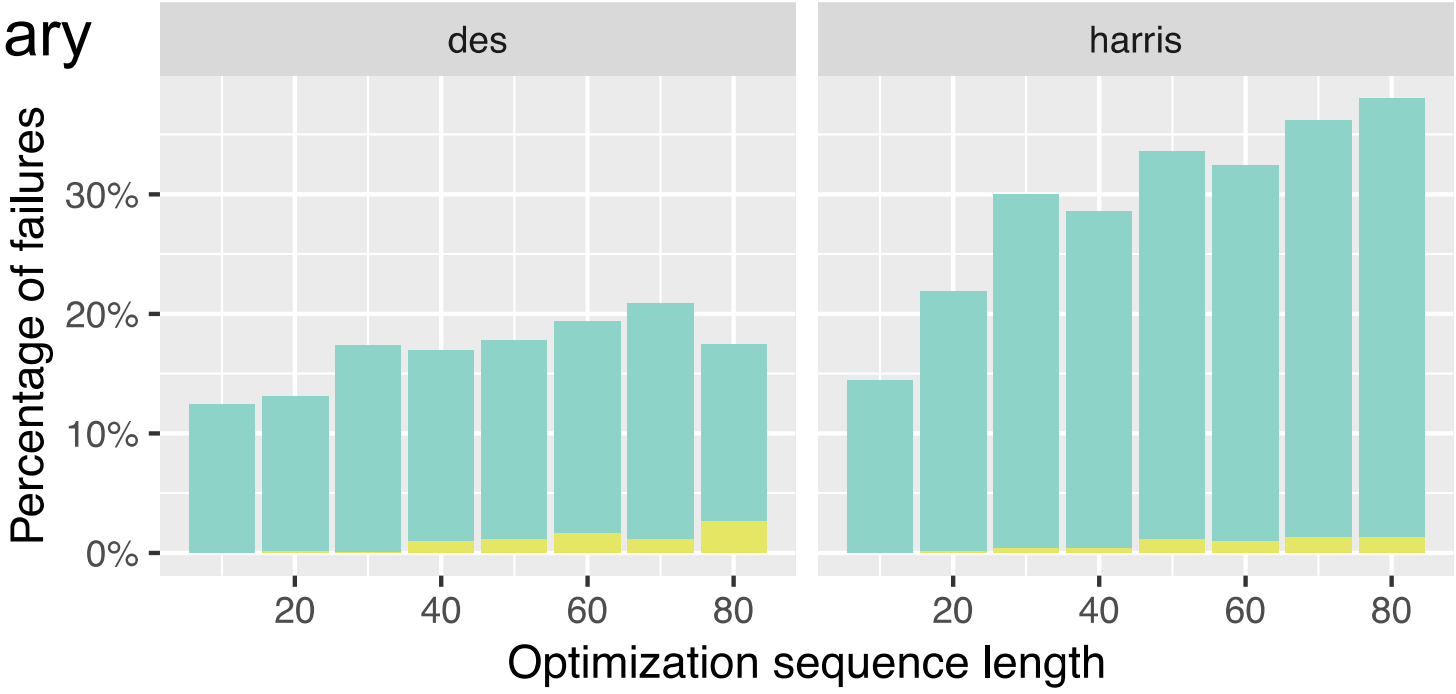
- Can render the binary non –analyzable
- Can make the compiler crash



Impact on analyzability

Different Optimisations:

- Can render the binary non –analyzable
- Can make the compiler crash

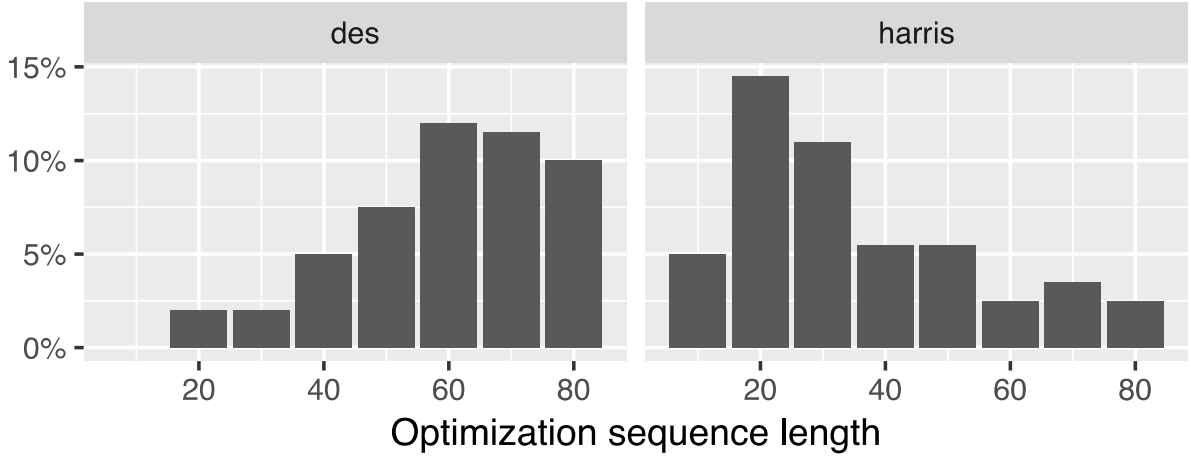


Optimisations can break analyzability

Impact on different applications

Different optimisations:

- Varying behavior on different applications



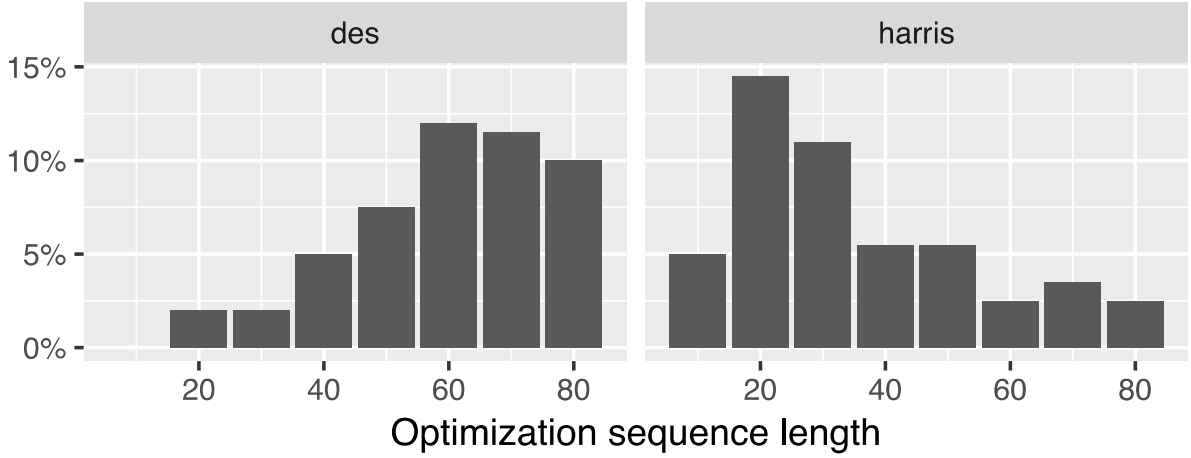
Best sequence:

Des	**-loop-rotate -sink -indvars -partial-inliner -loop-simplify -always-inline -deadargelim -mem2reg -loop-unroll -partial-inliner -lcssa -simplifycfg -lowerinvoke -inline -loop-reduce -loop-simplify -strip-dead-prototypes -loop-unswitch -loop-deletion -die -reassociate -constmerge -loop-deletion -lowerinvoke -ipconstprop -globalopt -mergereturn**
Harris	**-always-inline -always-inline -functionattrs -prune-eh -loop-simplify -dce -strip-dead-debug-info -codegenprepare -mem2reg -loop-rotate -partial-inliner -simplifycfg -functionattrs -adce -prune-eh -globaldce -loop-unswitch -strip-nondebug -strip-nondebug -simplifycfg -functionattrs -loweratomic -constmerge -globaldce -strip-nondebug -globaldce -constprop -loop-deletion -lcssa -instcombine**

Impact on different applications

Different optimisations:

- Varying behavior on different applications



Best sequence:

Des	**-loop-rotate -sink -indvars -partial-inliner -loop-simplify -always-inline -deadargelim -mem2reg -loop-unroll -partial-inliner -lcssa -simplifycfg -lowerinvoke -inline -loop-reduce -loop-simplify -strip-dead-prototypes -loop-unswitch -loop-deletion -die -reassociate -constmerge -loop-deletion -lowerinvoke -ipconstprop -globalopt -mergereturn**
Harris	**-always-inline -always-inline -functionattrs -prune-eh -loop-simplify -dce -strip-dead-debug-info -codegenprepare -mem2reg -loop-rotate -partial-inliner -simplifycfg -functionattrs -adce -prune-eh -globaldce -loop-unswitch -strip-nondebug -strip-nondebug -simplifycfg -functionattrs -loweratomic -constmerge -globaldce -strip-nondebug -globaldce -constprop -loop-deletion -lcssa -instcombine**

Optimisations behavior is application dependent

Motivation

- **Mainstream compiler optimisations sequences:**
  - Rich set of optimisation options
  - *Positive, Neutral, Negative* impact on WCET
  - Impact is application-dependent
  - Vast search space to find the best sequence

Motivation

- **Mainstream compiler optimisations sequences:**
  - Rich set of optimisation options
  - *Positive, Neutral, Negative* impact on WCET
  - Impact is application-dependent
  - Vast search space to find the best sequence

- **Our Idea: Iterative compilation + Machine learning**
  - First systematic attempt for WCET

Outline

- Motivation
- Our approach:
  - Iterative compilation
  - Association-based learning strategy
- Evaluation

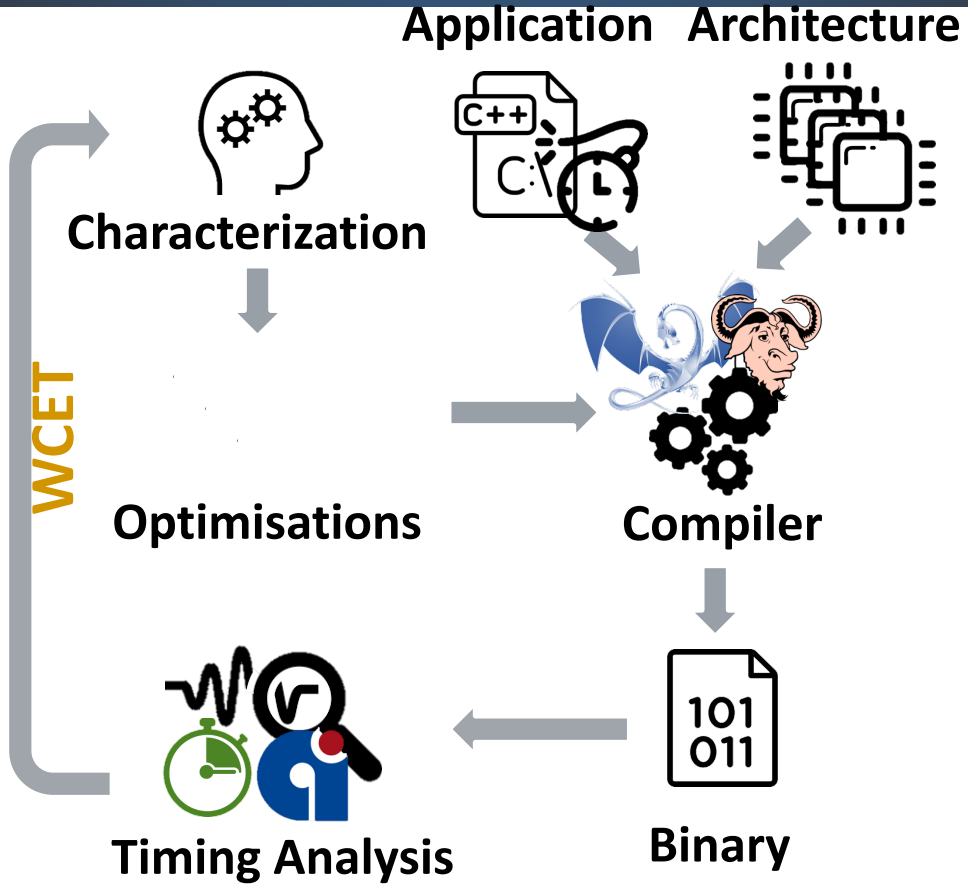
Iterative Compilation

- **Basic idea:**
  - Compile with different options
  - Evaluate the produced artifact
  - Iterate until result is satisfactory

- **Has been effectively used in best-effort systems**

Iterative Compilation

- **Basic idea:**
  - Compile with different options
  - Evaluate the produced artifact
  - Iterate until result is satisfactory
- **Has been effectively used in best-effort systems**



Pass Characterization Strategy

- **Sequences of optimization passes, e.g. *ABC***
  - A pass can appear multiple times, e.g. *ABA*
  - Sequence space is infinite

Pass Characterization Strategy

- **Sequences of optimization passes, e.g. *ABC***
  - A pass can appear multiple times, e.g. *ABA*
  - Sequence space is infinite

- **Given two sequences S ⊆ S'**
  - S' is *positive* if $WCET_{S'} < WCET_S$
  - S' is *neutral* if $WCET_{S'} = WCET_S$
  - S' is *negative* if $WCET_{S'} > WCET_S$

Pass Characterization Strategy

- **Sequences of optimization passes, e.g. *ABC***
  - A pass can appear multiple times, e.g. *ABA*
  - Sequence space is infinite

- **Given two sequences S ⊆ S'**
  - S' is *positive* if $WCET_{S'} < WCET_S$
  - S' is *neutral* if $WCET_{S'} = WCET_S$
  - S' is *negative* if $WCET_{S'} > WCET_S$

- **Pass *p* characterization**
  - $p \in S \cap S'$: same as S
  - $p \notin S \cap S'$: same as S'

Pass Characterization Strategy

- **Sequences of optimization passes, e.g. *ABC***
  - A pass can appear multiple times, e.g. *ABA*
  - Sequence space is infinite

- **Given two sequences S ⊆ S'**
  - S' is *positive* if $WCET_{S'} < WCET_S$
  - S' is *neutral* if $WCET_{S'} = WCET_S$
  - S' is *negative* if $WCET_{S'} > WCET_S$

- **Pass *p* characterization**
  - $p \in S \cap S'$: same as S
  - $p \notin S \cap S'$: same as S'

{ A , B , C }

{ A^+, C^+ }

$WCET_{ABC} = WCET_{AC}$

Pass Characterization Strategy

- **Sequences of optimization passes, e.g. *ABC***
  - A pass can appear multiple times, e.g. *ABA*
  - Sequence space is infinite

- **Given two sequences S ⊆ S'**
  - S' is *positive* if $WCET_{S'} < WCET_S$
  - S' is *neutral* if $WCET_{S'} = WCET_S$
  - S' is *negative* if $WCET_{S'} > WCET_S$

- **Pass *p* characterization**
  - $p \in S \cap S'$: same as S
  - $p \notin S \cap S'$: same as S'

$\{ A^+, B^\circ, C^+ \}$

$\{ A^+, C^+ \}$

$WCET_{ABC} = WCET_{AC}$

Pass Selection Strategy

- **Assign weights to passes**
  - According to the characterisation

$$W_{pass} = N^{+} / N^{\circ} * 85\% + 15\%$$

Pass Selection Strategy

- **Assign weights to passes**
  - According to the characterisation

$$W_{pass} = N^{+} / N^{\circ} * 85\% + 15\%$$

- **Intuition**
  - Frequently positive passes should be selected
  - Neutral pass should be avoided

Pass Selection Strategy

- **Assign weights to passes**
  - According to the characterisation

$$W_{pass} = N^{+} / N^{\circ} * 85\% + 15\%$$

- **Intuition**
  - Frequently positive passes should be selected
  - Neutral pass should be avoided
- ***Selection***
  - *Random size of sequence*
  - *Weighted random for the sequences passes*

Characterization Strategy: Example

- **1st Sequence: *ABC***
  - $WCET_{ABC} < WCET_{\varnothing}$

A	B	C	D
0.15	0.15	0.15	0.15

{ A , B , C , D }

{ A , B , C } { A , B , D } { A , C , D } { B , C , D }

{ A , B } { A , C } { A , D } { B , C } { B , D } { C , D }

{ A } { B } { C } { D }

{ Ø } $WCET_{\varnothing}$

Characterization Strategy: Example

- **1st Sequence: *ABC***
  - $WCET_{ABC} < WCET_{\varnothing}$

A	B	C	D
0.15	0.15	0.15	0.15

{ A , B , C , D }

{ A , B , D } { A , C , D } { B , C , D }

{ A+, B+, C+ }

{ A , B } { A , C } { A , D } { B , C } { B , D } { C , D }

$WCET_{ABC} < WCET_{\varnothing}$

{ A } { B } { C } { D }

{ Ø } $WCET_{\varnothing}$

Characterization Strategy: Example

- **1st Sequence: *ABC***
  - $WCET_{ABC} < WCET_{\varnothing}$

A	B	C	D
1.00	1.00	1.00	0.15

{ A , B , C , D }

{ A , B , D } { A , C , D } { B , C , D }

{ A+, B+, C+ }

{ A , B } { A , C } { A , D } { B , C } { B , D } { C , D }

$WCET_{ABC} < WCET_{\varnothing}$

{ A } { B } { C } { D }

{ Ø } $WCET_{\varnothing}$

Characterization Strategy: Example

- **1st Sequence: *ABC***
  - $WCET_{ABC} < WCET_{\varnothing}$
- **Next Sequence: *ACD***
  - $WCET_{ACD} > WCET_{\varnothing}$
  - Common flags *AC*

A	B	C	D
1.00	1.00	1.00	0.15

{ A , B , C , D }

{ A^+, B^+, C^+ } { A , B , D } { A , C , D } { B , C , D }

{ A , B } { A , C } { A , D } { B , C } { B , D } { C , D }

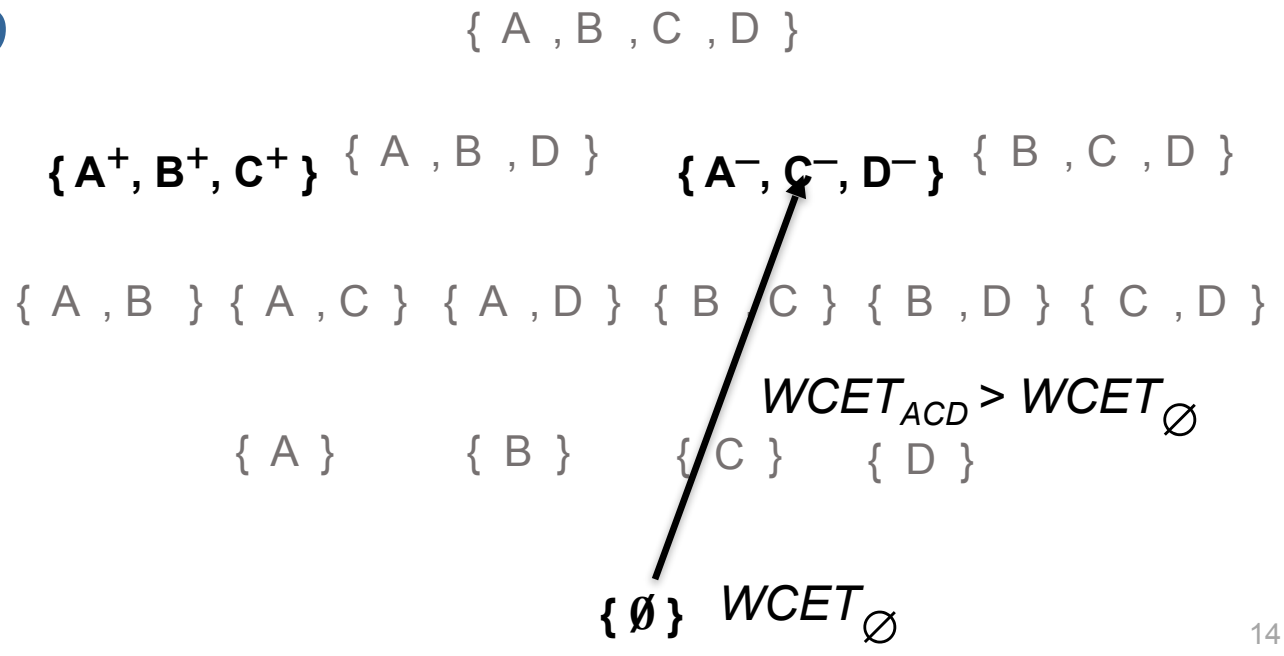
{ A } { B } { C } { D }

{ Ø } $WCET_{\varnothing}$

Characterization Strategy: Example

- **1st Sequence: *ABC***
  - $WCET_{ABC} < WCET_{\varnothing}$
- **Next Sequence: *ACD***
  - $WCET_{ACD} > WCET_{\varnothing}$
  - Common flags *AC*

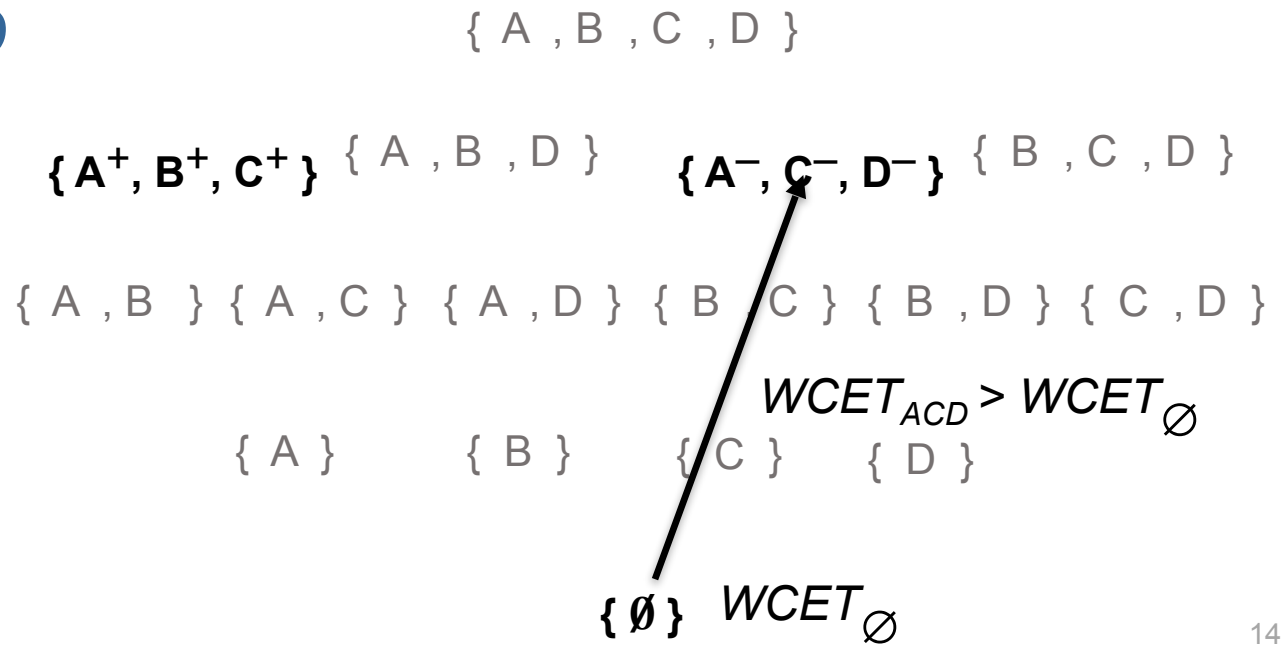
A	B	C	D
1.00	1.00	1.00	0.15



Characterization Strategy: Example

- **1st Sequence: *ABC***
  - $WCET_{ABC} < WCET_{\varnothing}$
- **Next Sequence: *ACD***
  - $WCET_{ACD} > WCET_{\varnothing}$
  - Common flags *AC*

A	B	C	D
1.00	1.00	1.00	0.15



Characterization Strategy: Example

- **1st Sequence: *ABC***
  - $WCET_{ABC} < WCET_{\varnothing}$
- **Next Sequence: *ACD***
  - $WCET_{ACD} > WCET_{\varnothing}$
  - Common flags *AC*
- **Next Sequence: *AC***
  - $WCET_{AC} < WCET_{\varnothing}$
  - $WCET_{AC} < WCET_{ACD}$
  - $WCET_{ABC} = WCET_{AC}$

A	B	C	D
1.00	1.00	1.00	0.15

{ A , B , C , D }

{ A^+, B^+, C^+ } { A , B , D } **{ A^-, C^-, D^- }** { B , C , D }

{ A , B } { A , C } { A , D } { B , C } { B , D } { C , D }

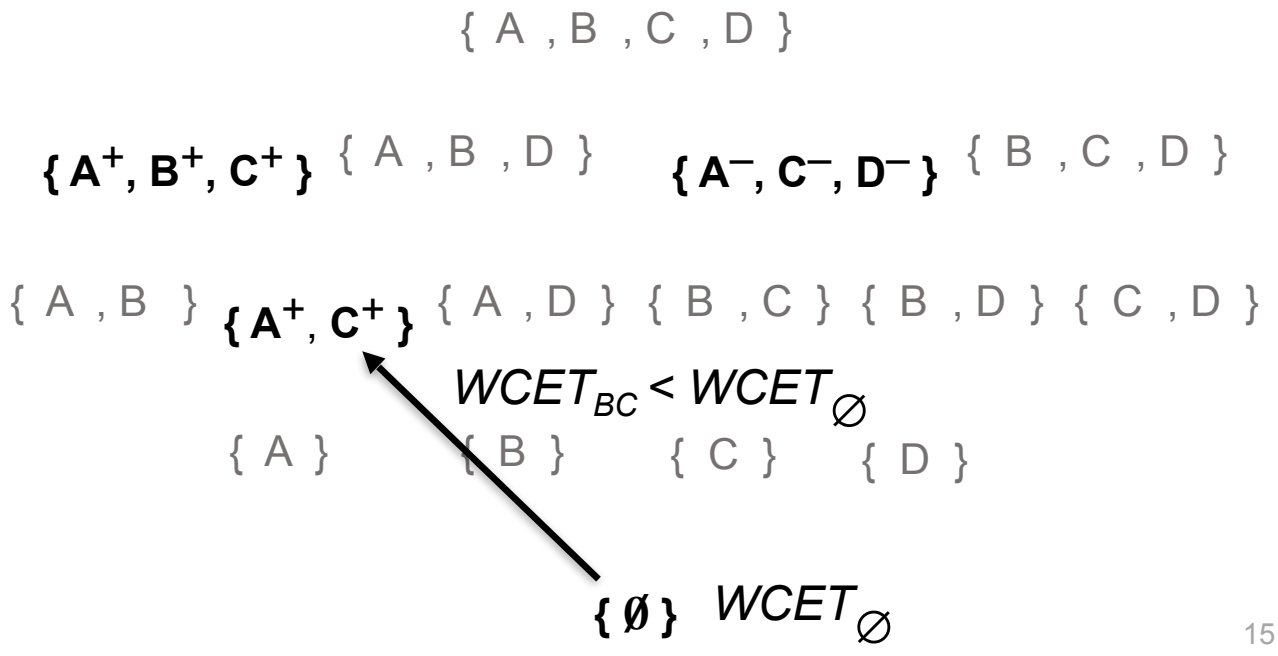
{ A } { B } { C } { D }

{ Ø } $WCET_{\varnothing}$

Characterization Strategy: Example

- **1st Sequence: *ABC***
  - $WCET_{ABC} < WCET_{\varnothing}$
- **Next Sequence: *ACD***
  - $WCET_{ACD} > WCET_{\varnothing}$
  - Common flags *AC*
- **Next Sequence: *AC***
  - $WCET_{AC} < WCET_{\varnothing}$
  - $WCET_{AC} < WCET_{ACD}$
  - $WCET_{ABC} = WCET_{AC}$

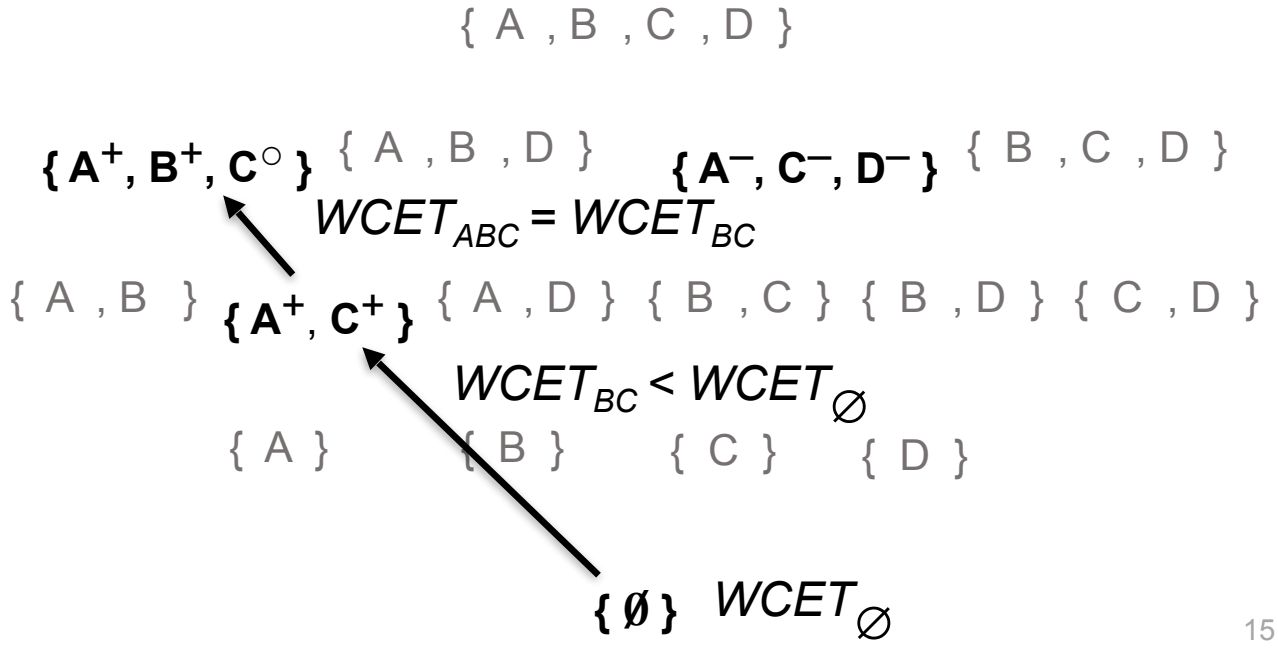
A	B	C	D
1.00	1.00	1.00	0.15



Characterization Strategy: Example

- **1st Sequence: *ABC***
  - $WCET_{ABC} < WCET_{\varnothing}$
- **Next Sequence: *ACD***
  - $WCET_{ACD} > WCET_{\varnothing}$
  - Common flags *AC*
- **Next Sequence: *AC***
  - $WCET_{AC} < WCET_{\varnothing}$
  - $WCET_{AC} < WCET_{ACD}$
  - $WCET_{ABC} = WCET_{AC}$

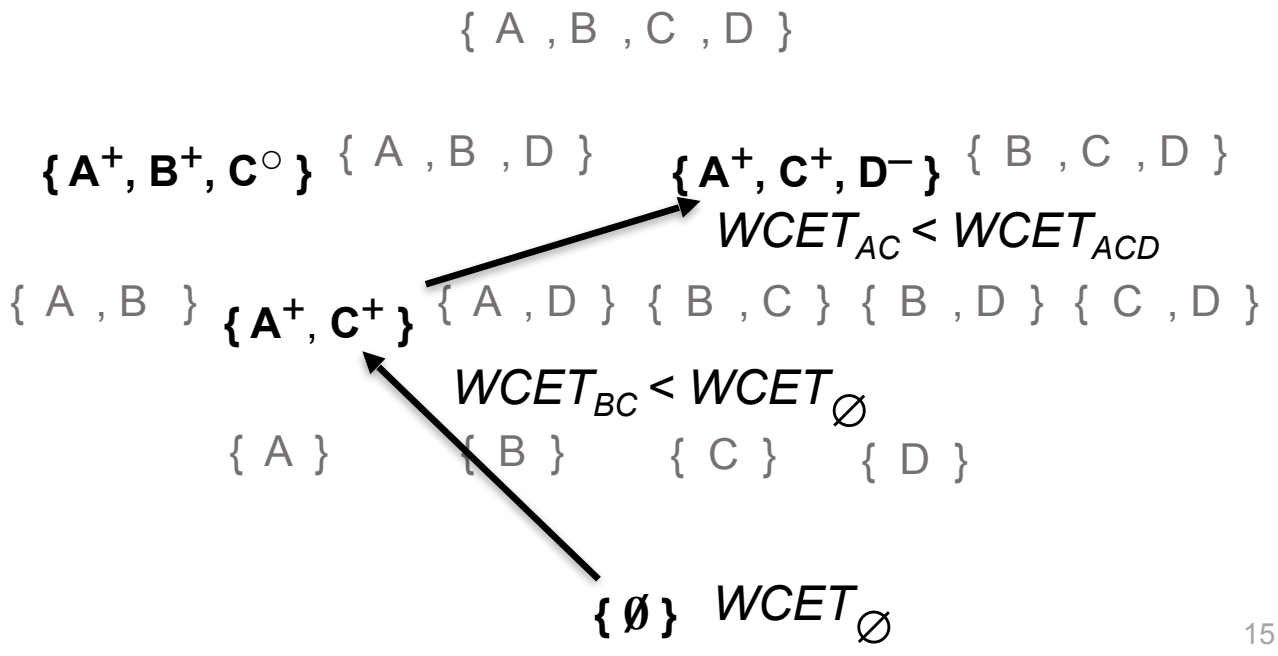
A	B	C	D
1.00	1.00	1.00	0.15



Characterization Strategy: Example

- **1st Sequence: *ABC***
  - $WCET_{ABC} < WCET_{\varnothing}$
- **Next Sequence: *ACD***
  - $WCET_{ACD} > WCET_{\varnothing}$
  - Common flags *AC*
- **Next Sequence: *AC***
  - $WCET_{AC} < WCET_{\varnothing}$
  - $WCET_{AC} < WCET_{ACD}$
  - $WCET_{ABC} = WCET_{AC}$

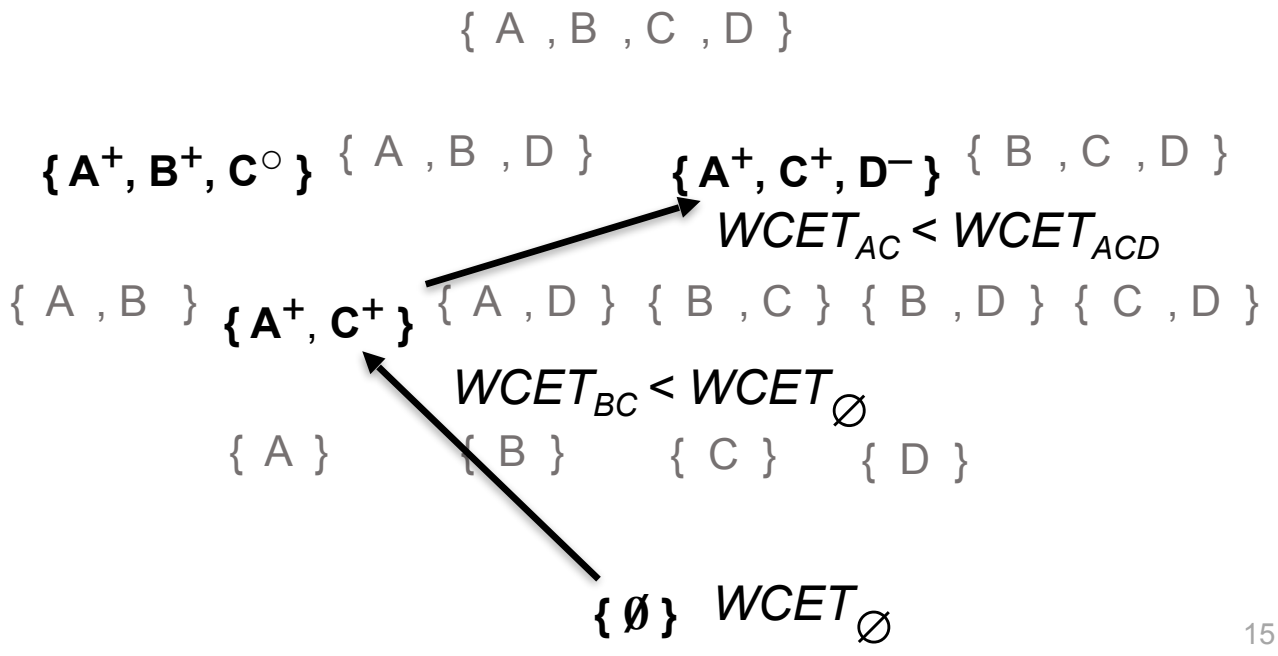
A	B	C	D
1.00	1.00	1.00	0.15



Characterization Strategy: Example

- **1st Sequence: *ABC***
  - $WCET_{ABC} < WCET_{\varnothing}$
- **Next Sequence: *ACD***
  - $WCET_{ACD} > WCET_{\varnothing}$
  - Common flags *AC*
- **Next Sequence: *AC***
  - $WCET_{AC} < WCET_{\varnothing}$
  - $WCET_{AC} < WCET_{ACD}$
  - $WCET_{ABC} = WCET_{AC}$

A	B	C	D
1.85	1.85	1.00	0.15



Pass Cleaning

- **A pass may be mischaracterized**
  - Pass *C* was characterized initially as *positive,* but then as neutral.

- **Iteratively remove passes that do not improve WCET**
  - Known technique in iterative compilation

Cleaning: Example

- **1st Sequence: *ABC***
  - $WCET_{ABC} < WCET_{\varnothing}$

A	B	C	D
0.15	0.15	0.15	0.15

{ A , B , C , D }

{ A^+, B^+, C^+ } { A , B , D } { A , C , D } { B , C , D }

{ A , B } { A , C } { A , D } { B , C } { B , D } { C , D }

{ A } { B } { C } { D }

{ Ø } $WCET_{\varnothing}$

Cleaning: Example

- **1st Sequence: *ABC***
  - $WCET_{ABC} < WCET_{\varnothing}$
- **Next Sequence: *AC***
  - $WCET_{AC} = WCET_{ABC}$

A	B	C	D
0.15	0.15	0.15	0.15

$\{A, B, C, D\}$

$\{A^+, B^\circ, C^+\}$ $\{A, B, D\}$ $\{A, C, D\}$ $\{B, C, D\}$

$\{A, B\}$ $\{A^+, C^+\}$ $\{A, D\}$ $\{B, C\}$ $\{B, D\}$ $\{C, D\}$

$\{A\}$ $\{B\}$ $\{C\}$ $\{D\}$

$\{\emptyset\}$ $WCET_{\varnothing}$

Cleaning: Example

- **1st Sequence: *ABC***
  - $WCET_{ABC} < WCET_{\varnothing}$
- **Next Sequence: *AC***
  - $WCET_{AC} = WCET_{ABC}$
- **Next Sequence: *A***
  - $WCET_{AC} < WCET_{A}$
  - $WCET_{A} < WCET_{\varnothing}$

A	B	C	D
0.15	0.15	0.15	0.15

{ A , B , C , D }

{ A⁺, B°, C⁺ } { A , B , D } { A , C , D } { B , C , D }

{ A , B } **{ A⁺, C⁺ }** { A , D } { B , C } { B , D } { C , D }

{ A⁺} { B } { C } { D }

{ Ø } $WCET_{\varnothing}$

Cleaning: Example

- **1st Sequence: *ABC***
  - $WCET_{ABC} < WCET_{\varnothing}$
- **Next Sequence: *AC***
  - $WCET_{AC} = WCET_{ABC}$
- **Next Sequence: *A***
  - $WCET_{AC} < WCET_{A}$
  - $WCET_{A} < WCET_{\varnothing}$
- **Next Sequence: *C***
  - $WCET_{AC} < WCET_{C}$
  - $WCET_{C} < WCET_{\varnothing}$

A	B	C	D
0.15	0.15	0.15	0.15

{ A , B , C , D }

{ A^+, B°, C^+ } { A , B , D } { A , C , D } { B , C , D }

{ A , B } **{ A^+, C^+ }** { A , D } { B , C } { B , D } { C , D }

{ A^+} { B } **{ C^+}** { D }

{ Ø } $WCET_{\varnothing}$

Cleaning: Example

- **1st Sequence: *ABC***
  - $WCET_{ABC} < WCET_{\varnothing}$
- **Next Sequence: *AC***
  - $WCET_{AC} = WCET_{ABC}$
- **Next Sequence: *A***
  - $WCET_{AC} < WCET_{A}$
  - $WCET_{A} < WCET_{\varnothing}$
- **Next Sequence: *C***
  - $WCET_{AC} < WCET_{C}$
  - $WCET_{C} < WCET_{\varnothing}$

A	B	C	D
3.15	0.15	3.15	0.15

{ A , B , C , D }

$\{A^{+}, B^{\circ}, C^{+}\}$ { A , B , D } { A , C , D } { B , C , D }

{ A , B } $\{A^{+}, C^{+}\}$ { A , D } { B , C } { B , D } { C , D }

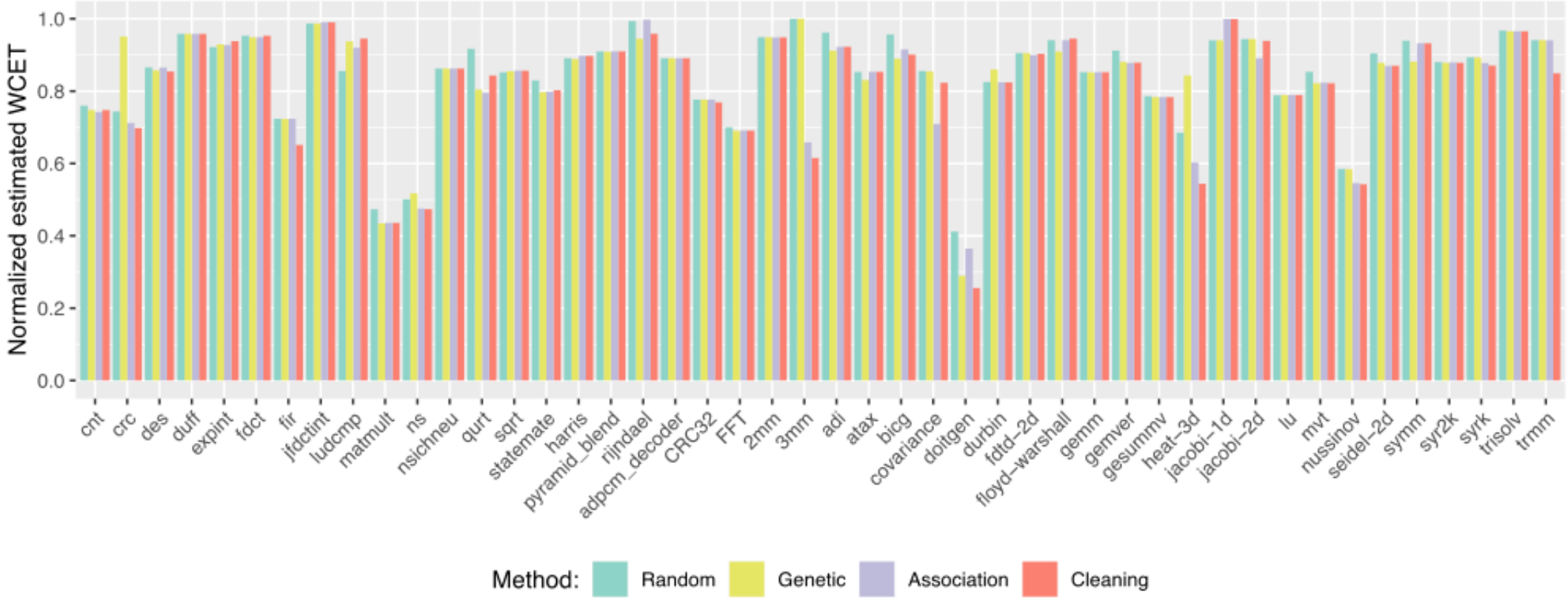
$\{A^{+}\}$ { B } $\{C^{+}\}$ { D }

$\{\varnothing\}$ $WCET_{\varnothing}$

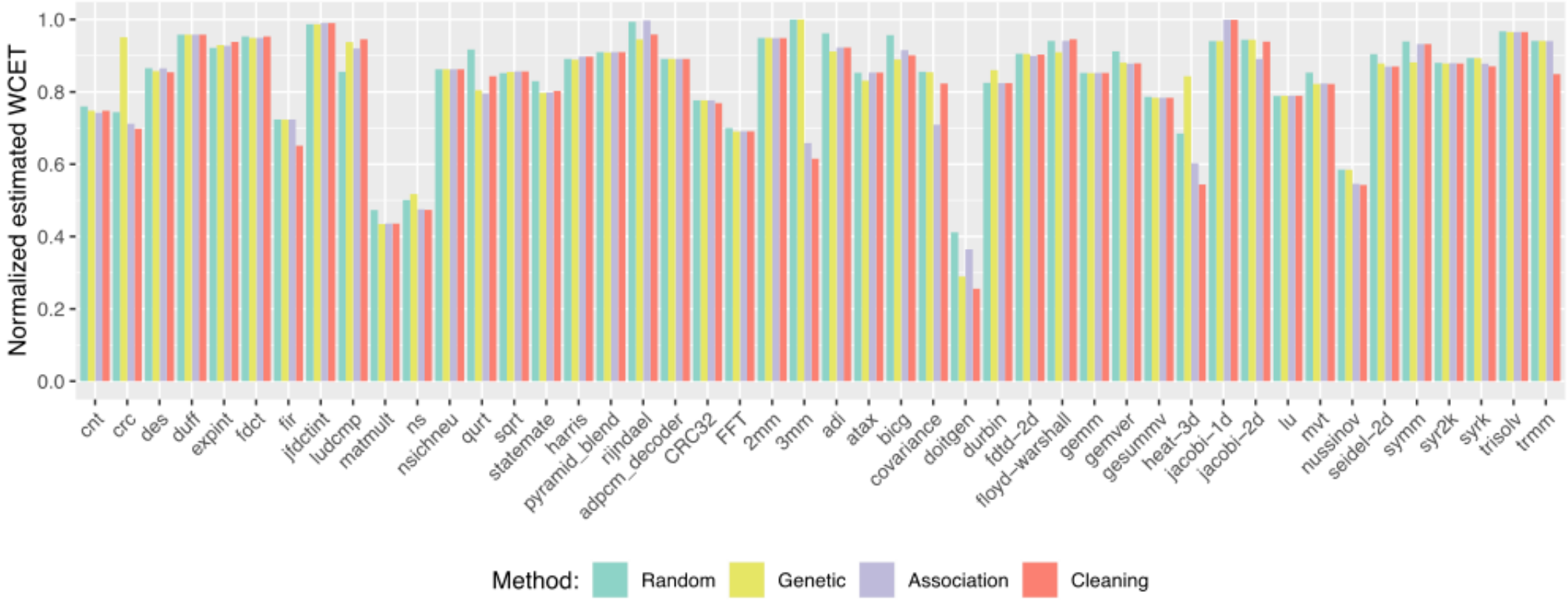
Evaluation

- **46 Benchmarks from 4 suites**
  - Malardalen, Polybench, MiBench, PolyMage
- **Compiled with LLVM 4.0 for LEON3 processor**
- **Static WCET analysis using aiT**
  - 16 KB 2-way instruction and data caches
- **Four selection methods for iterative compilation (1000 sequences)**
  - Random
  - Genetic algorithm
  - Association-based learning
  - Association-based learning with cleaning

Results Overview



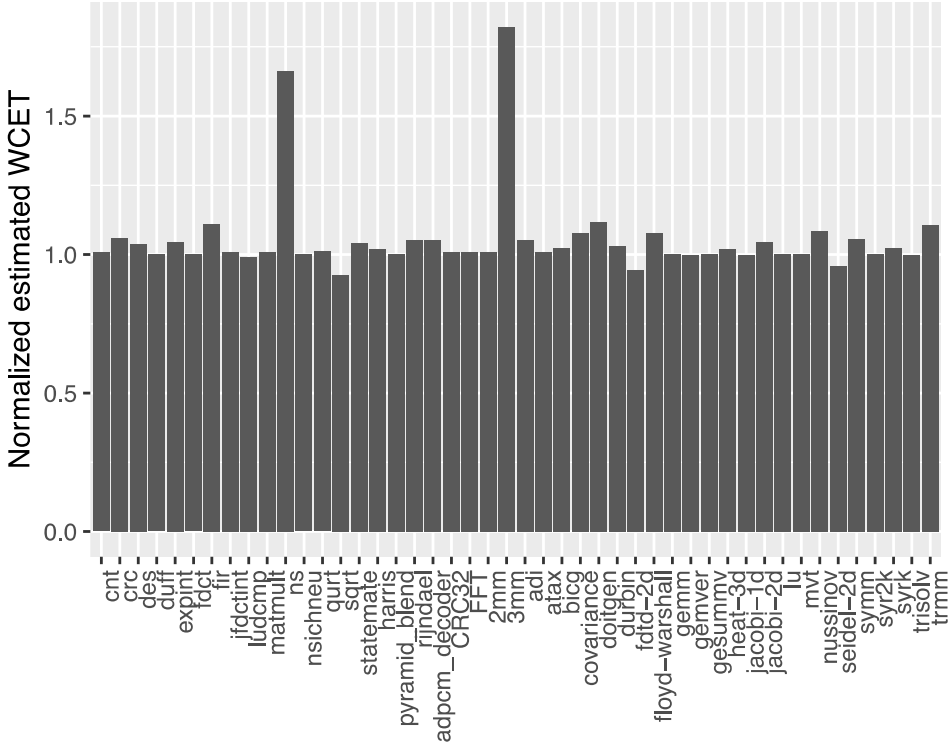
Results Overview



Iterative compilation can provide significant benefits

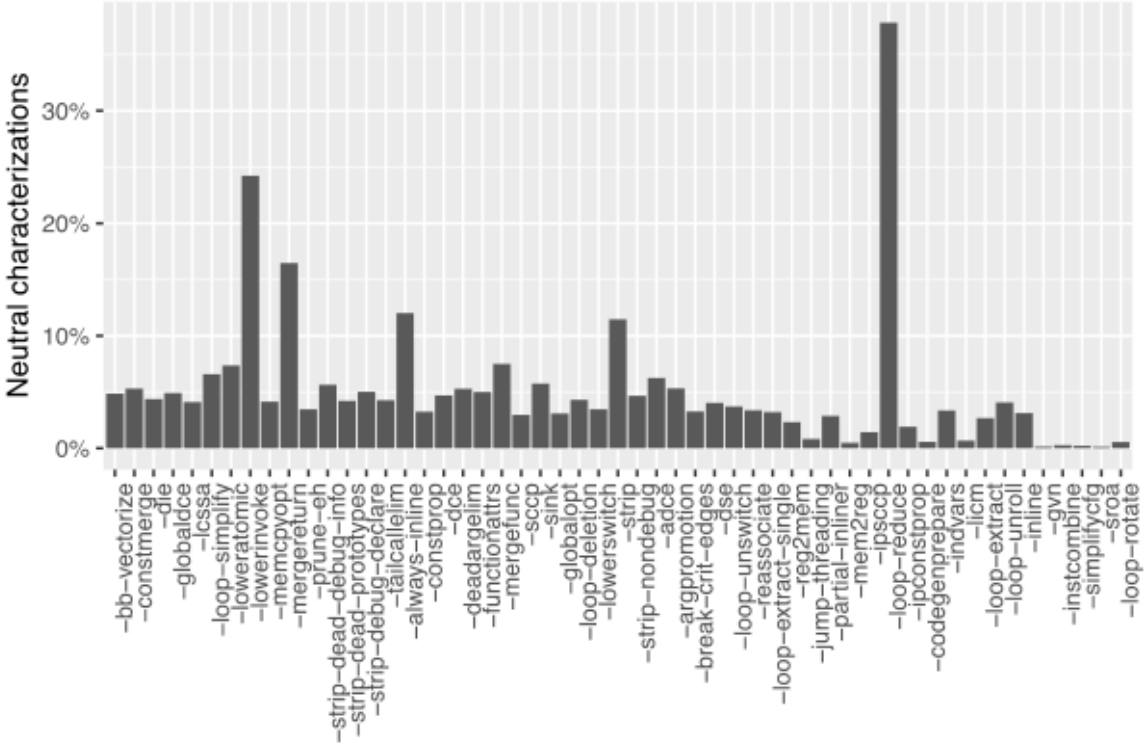
Assumptions cross validation

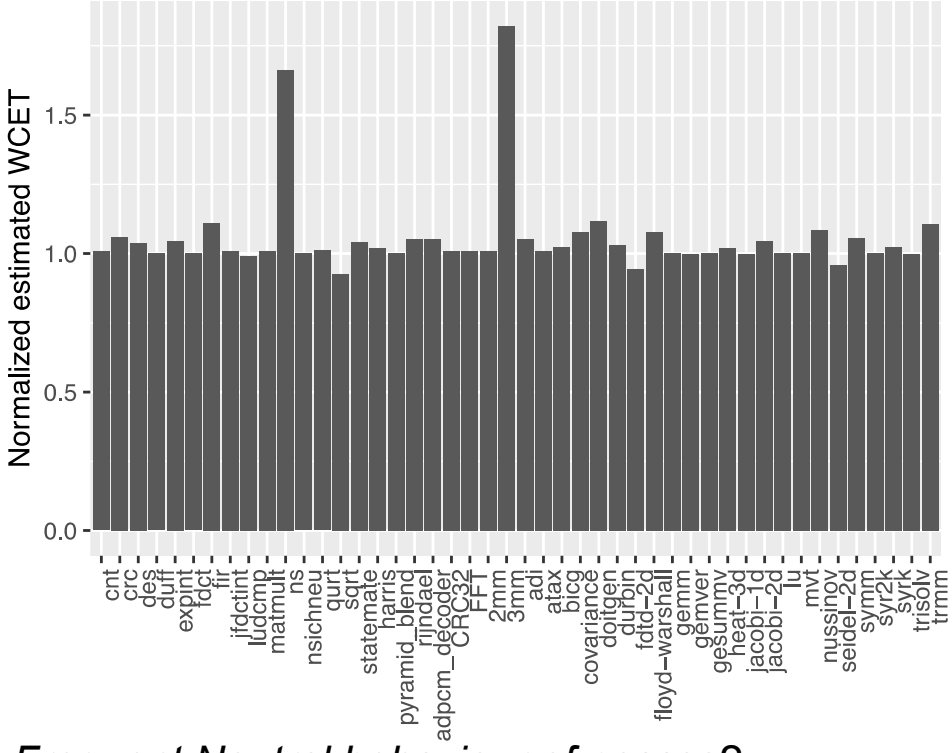
Application-dependent behaviour?
Apply sequences from other benchmarks



Assumptions cross validation

Application-dependent behaviour?
Apply sequences from other benchmarks

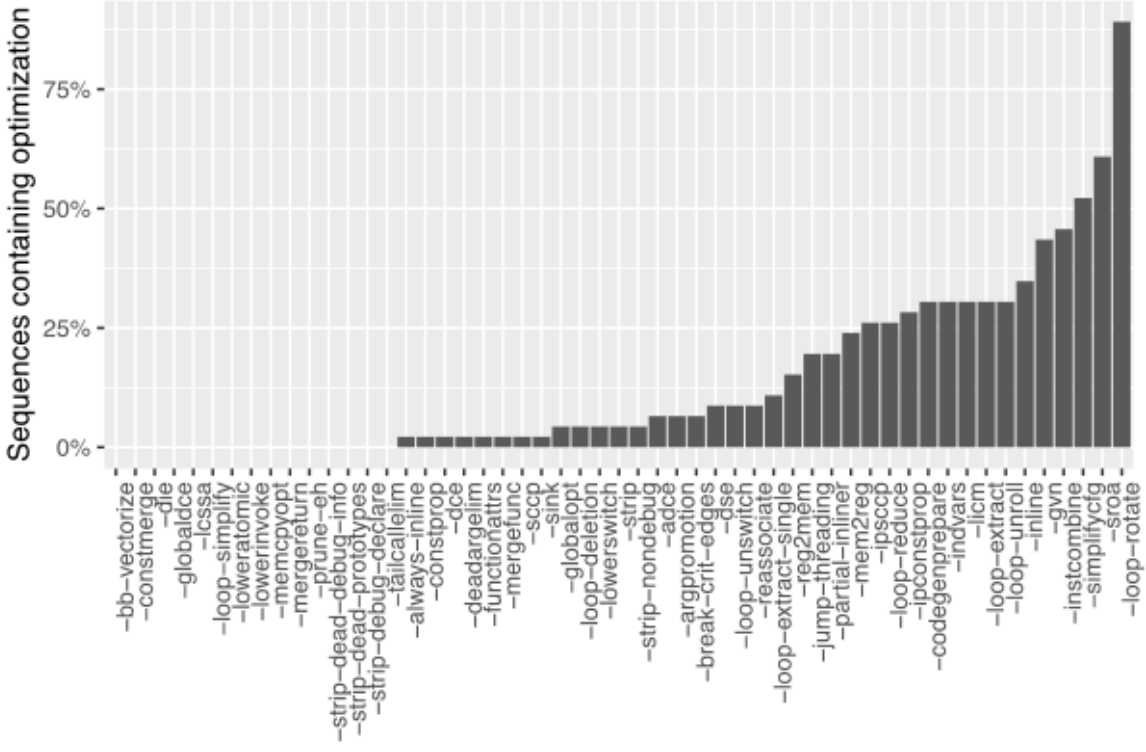




Frequent Neutral behaviour of passes?
Use existing characterisation

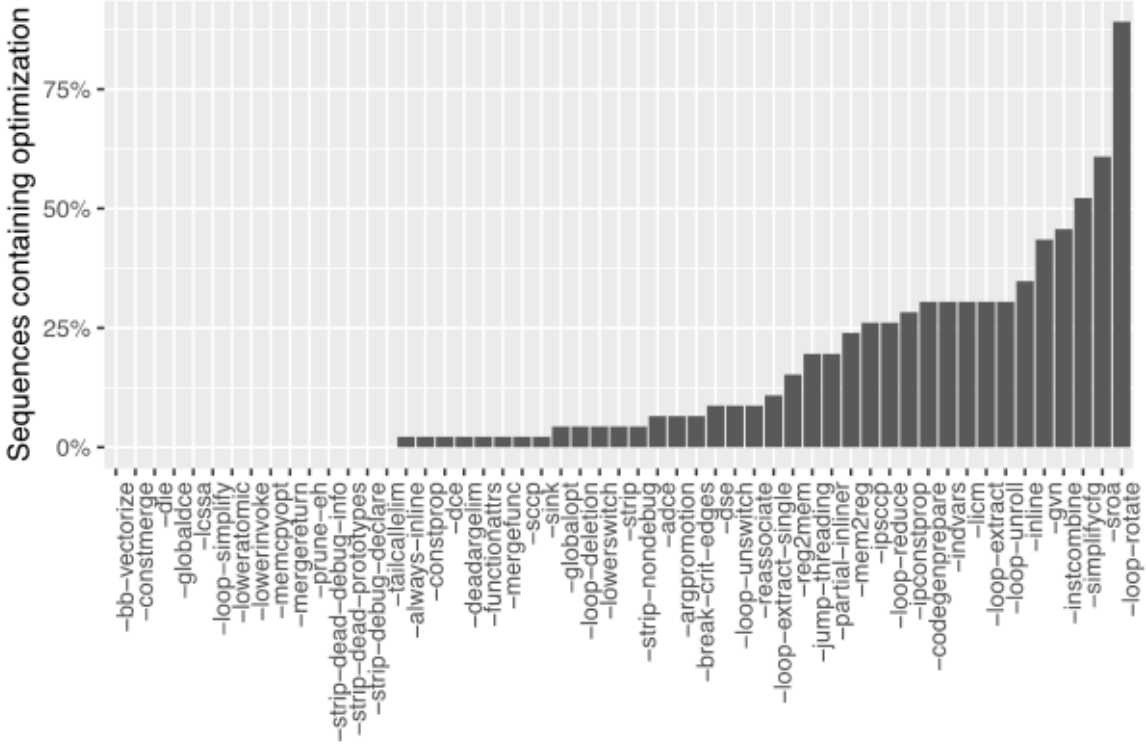
Pass / Sequence behavior

- **Pass frequency in best sequence**
  - For 1000 sequences / benchmark



Pass / Sequence behavior

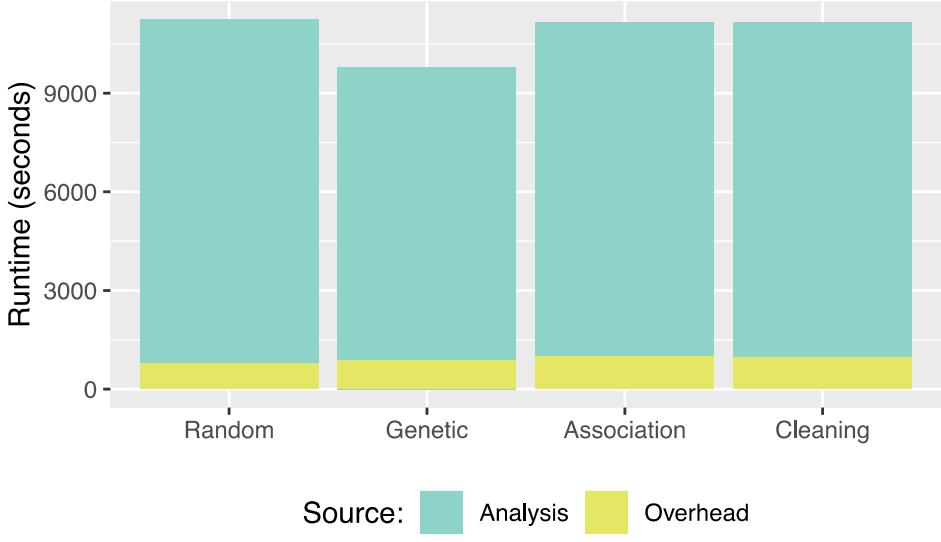
- **Pass frequency in best sequence**
  - For 1000 sequences / benchmark

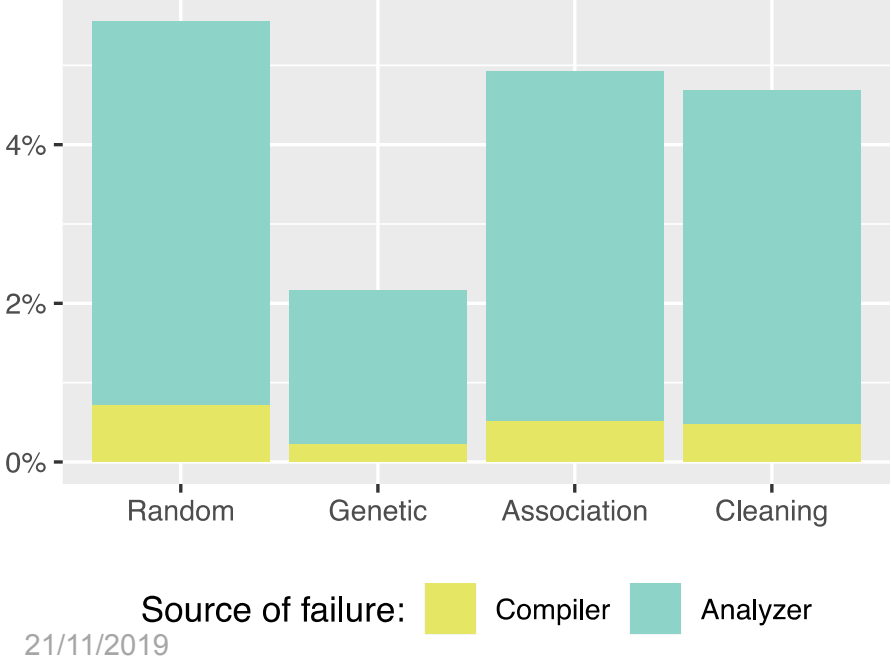


- **Best sequence length**
  - For 1000 sequences / benchmark

Runtime & Error susceptibility

- **Average runtime**
  - For 1000 sequences / benchmark

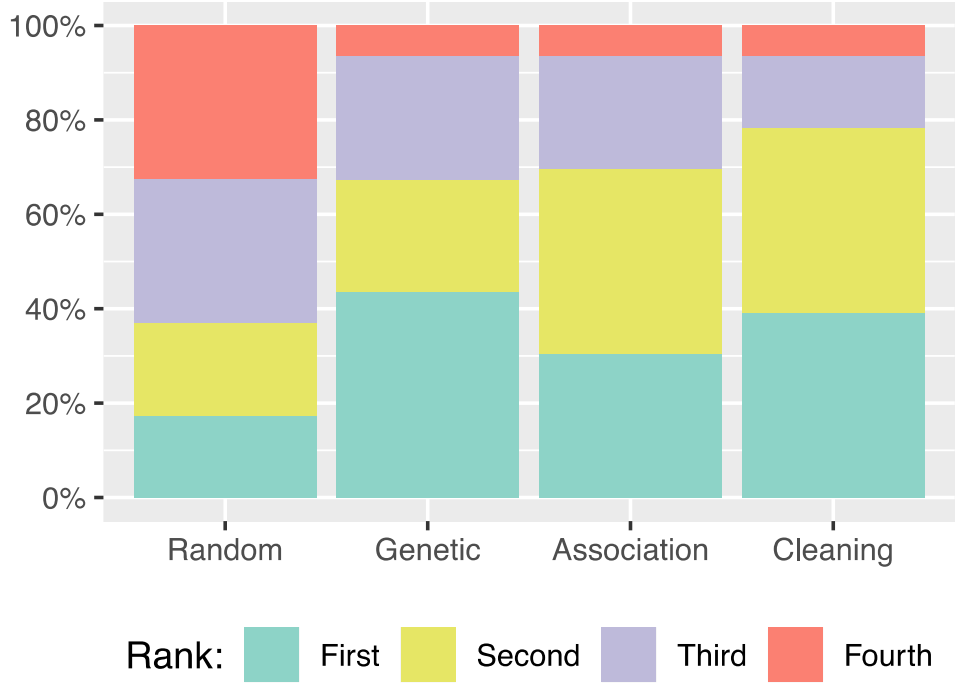




- **Average error rate**
  - For 1000 sequences / benchmark

Iterative Compilation: Results

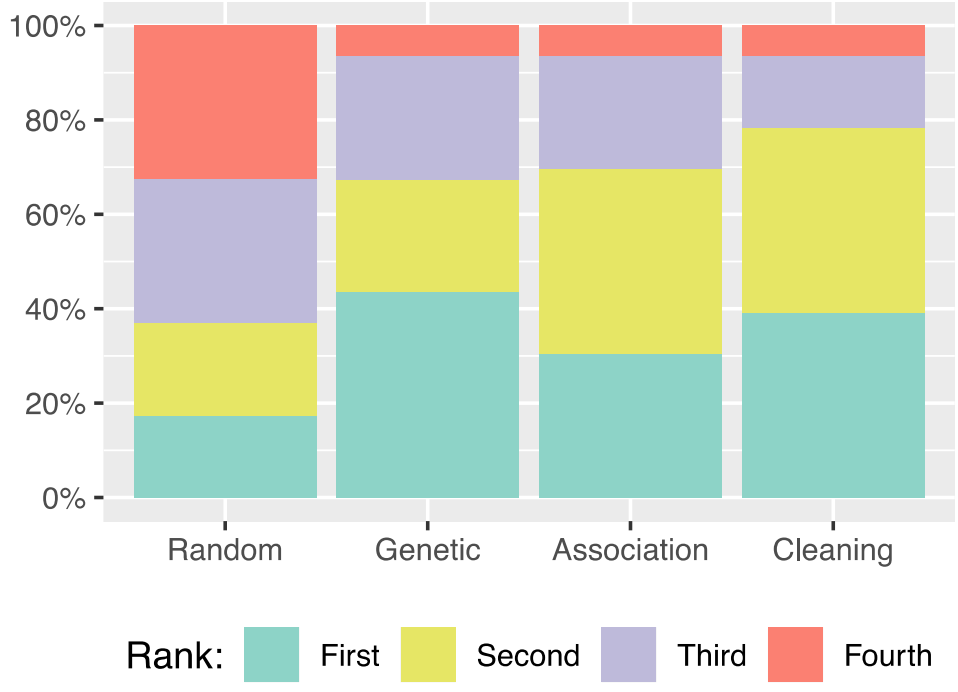
Characterization of optimisation passes using association-based learning



Iterative Compilation: Results

Characterization of optimisation passes using association-based learning

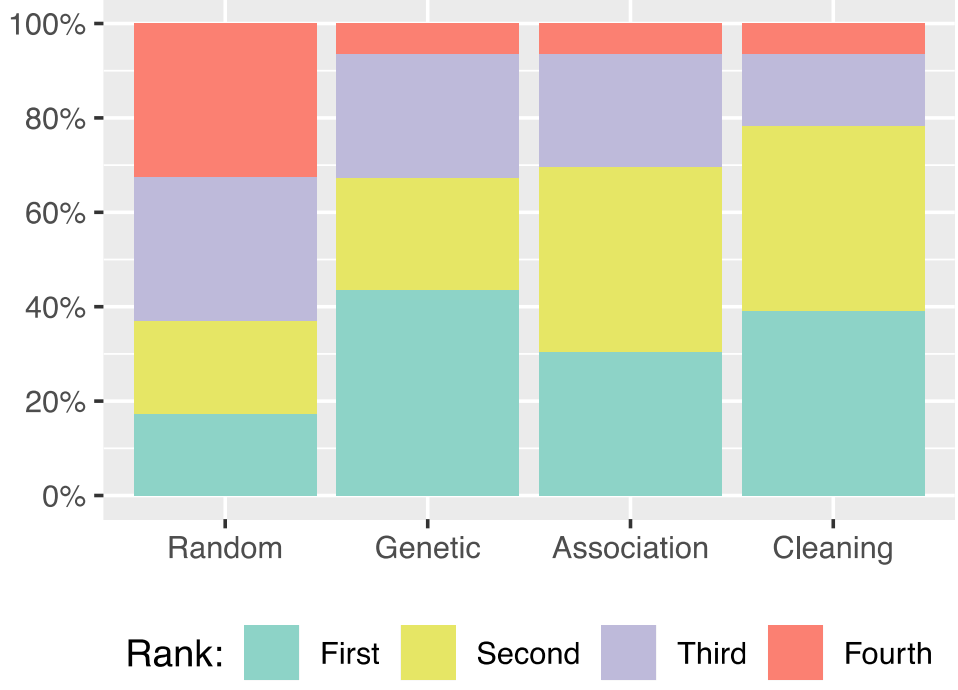
Method	(Geom.) Avg. WCET Improvement
Random	16.8%
Genetic	17.6%
Association	19.4%
Association + Cleaning	20.3%



Iterative Compilation: Results

Characterization of optimisation passes using association-based learning

Method	(Geom.) Avg. WCET Improvement
Random	16.8%
Genetic	17.6%
Association	19.4%
Association + Cleaning	20.3%



…thus providing significant WCET benefits!

Summary

- **Iterative compilation + machine learning**
  - Suitable for Static WCET analysis
- **Shown effectiveness of such approaches**
  - Proposed approach ~20% WCET reduction
- **In the paper:**
  - More technical details about parameter selection in aiT

Thank You

Questions?