Configuration-Aware Performance Analysis of Compile-Time Configurable HPC Systems

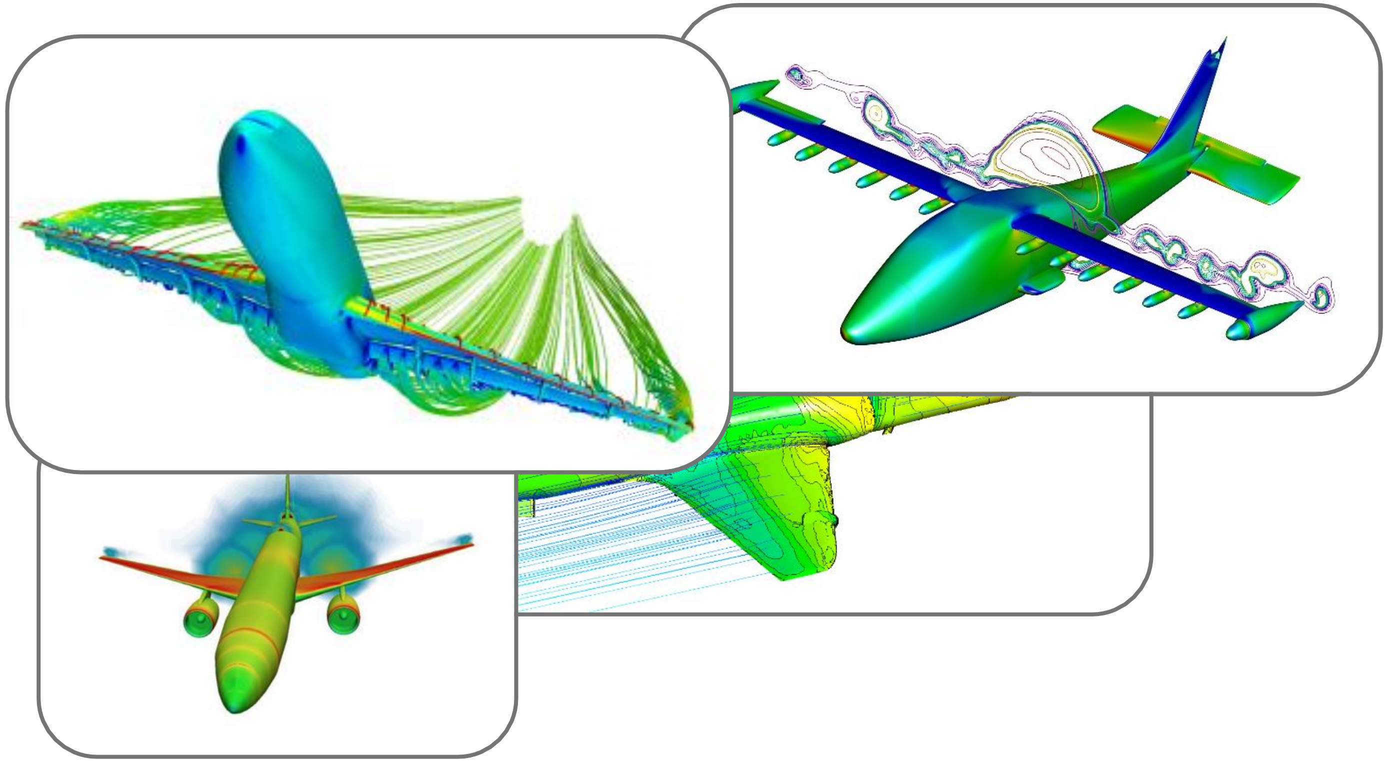
Lukas Abelt

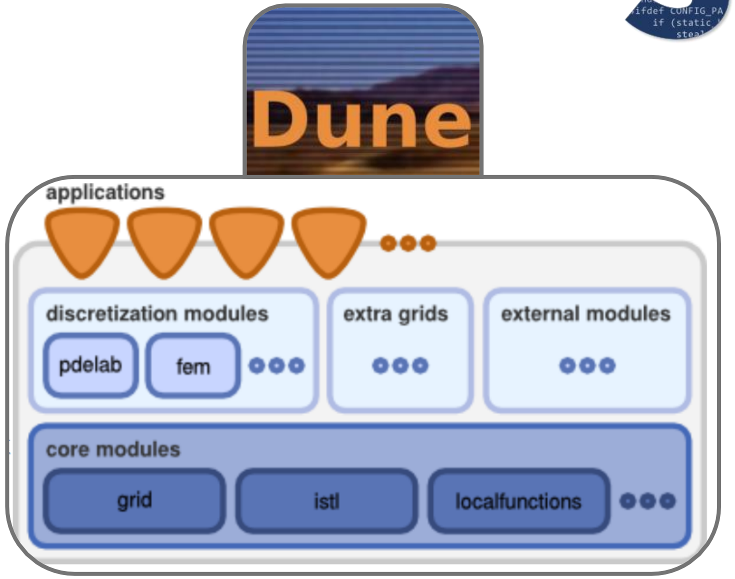
Florian Sattler

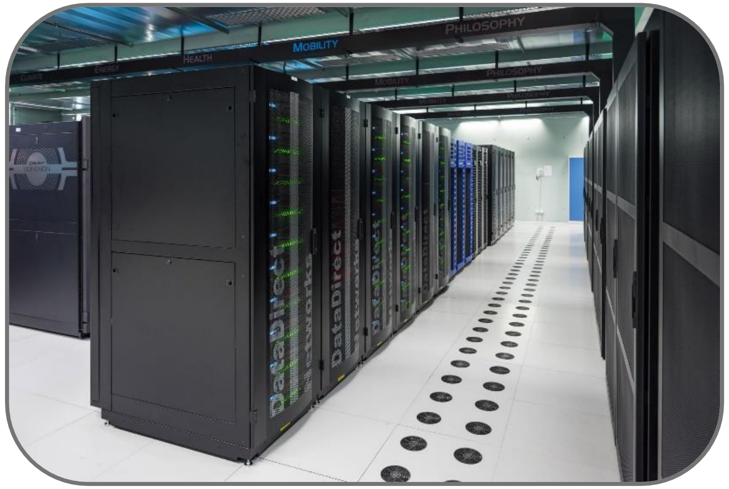
Prof. Sven Apel

Saarland Informatics Campus
Saarland University
FOSD Meeting 2024, Eindhoven

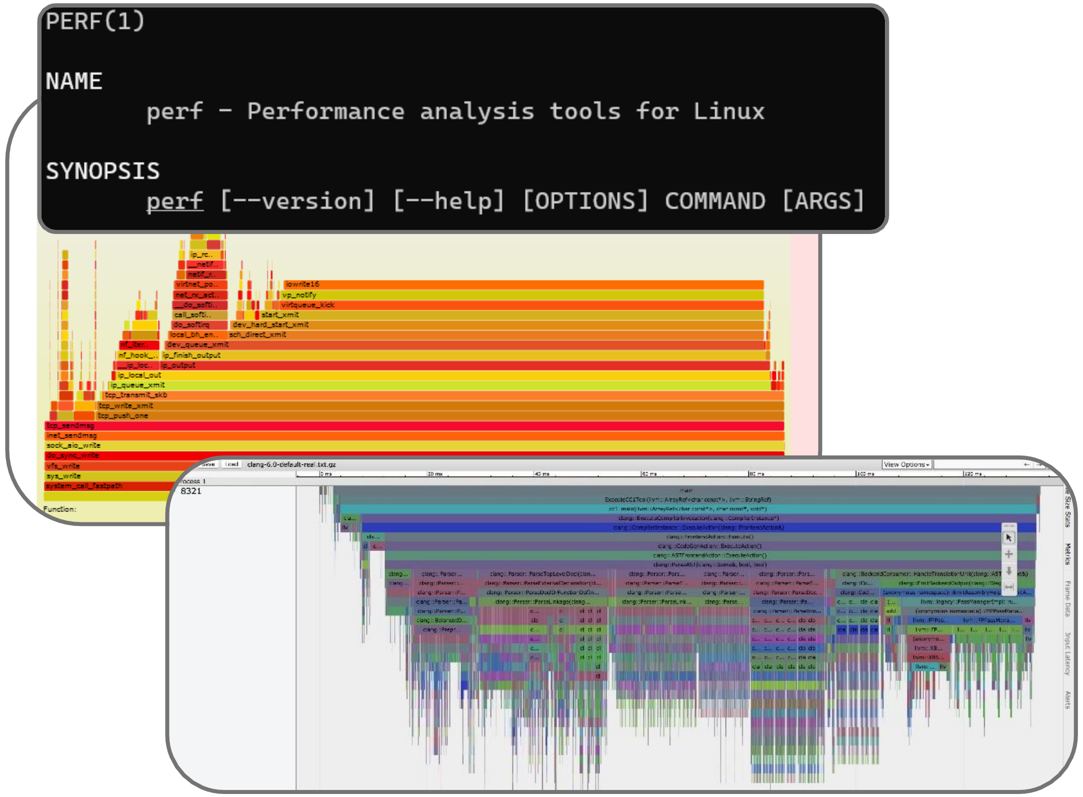
Motivation




Dune
applications
discretization modules
pdelab
fem
extra grids
external modules
core modules
grid
istl
localfunctions

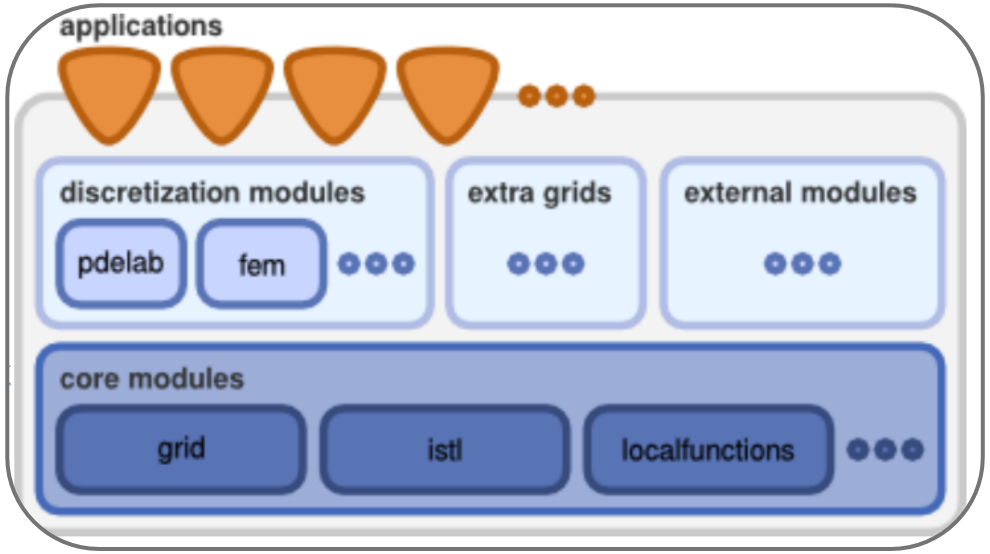

MOBILITY
PHILOSOPHY
DataDirect

Motivation



```
PERF(1)

NAME
       perf - Performance analysis tools for Linux

SYNOPSIS
       perf [--version] [--help] [OPTIONS] COMMAND [ARGS]
```

eBPF

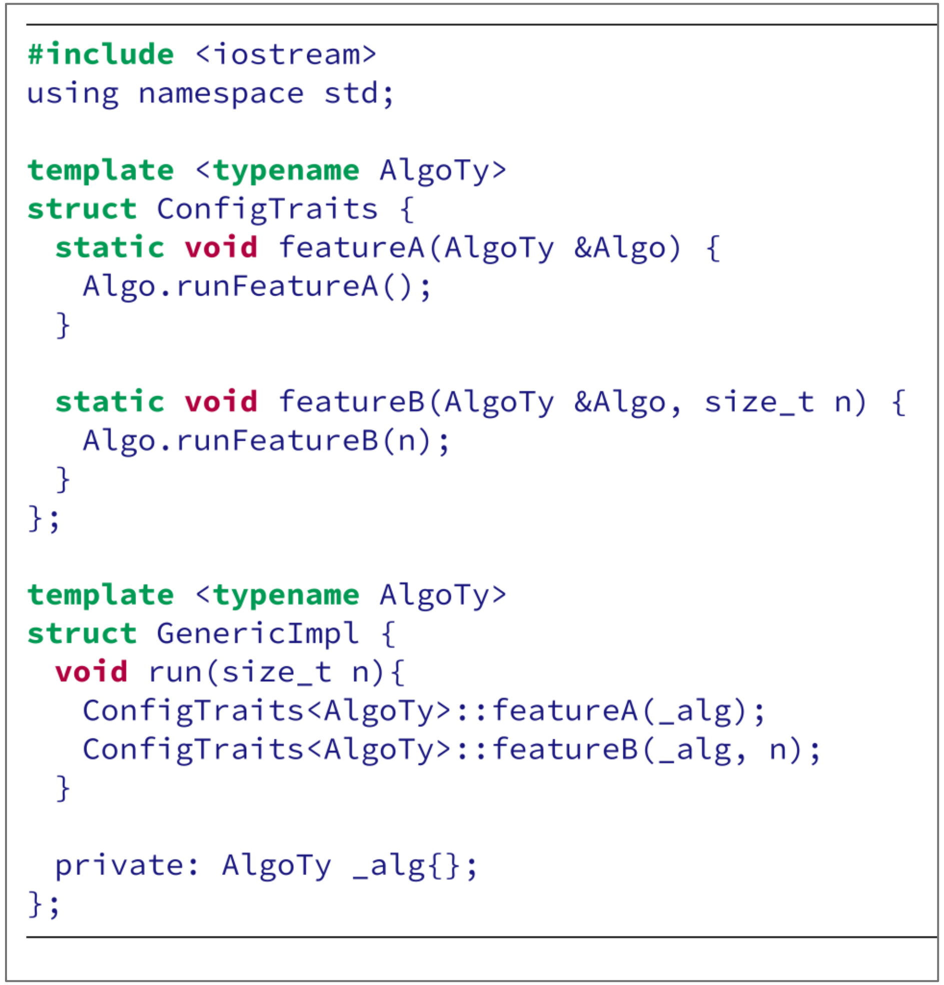


What are the capabilities and limitations of configuration-aware performance profilers?

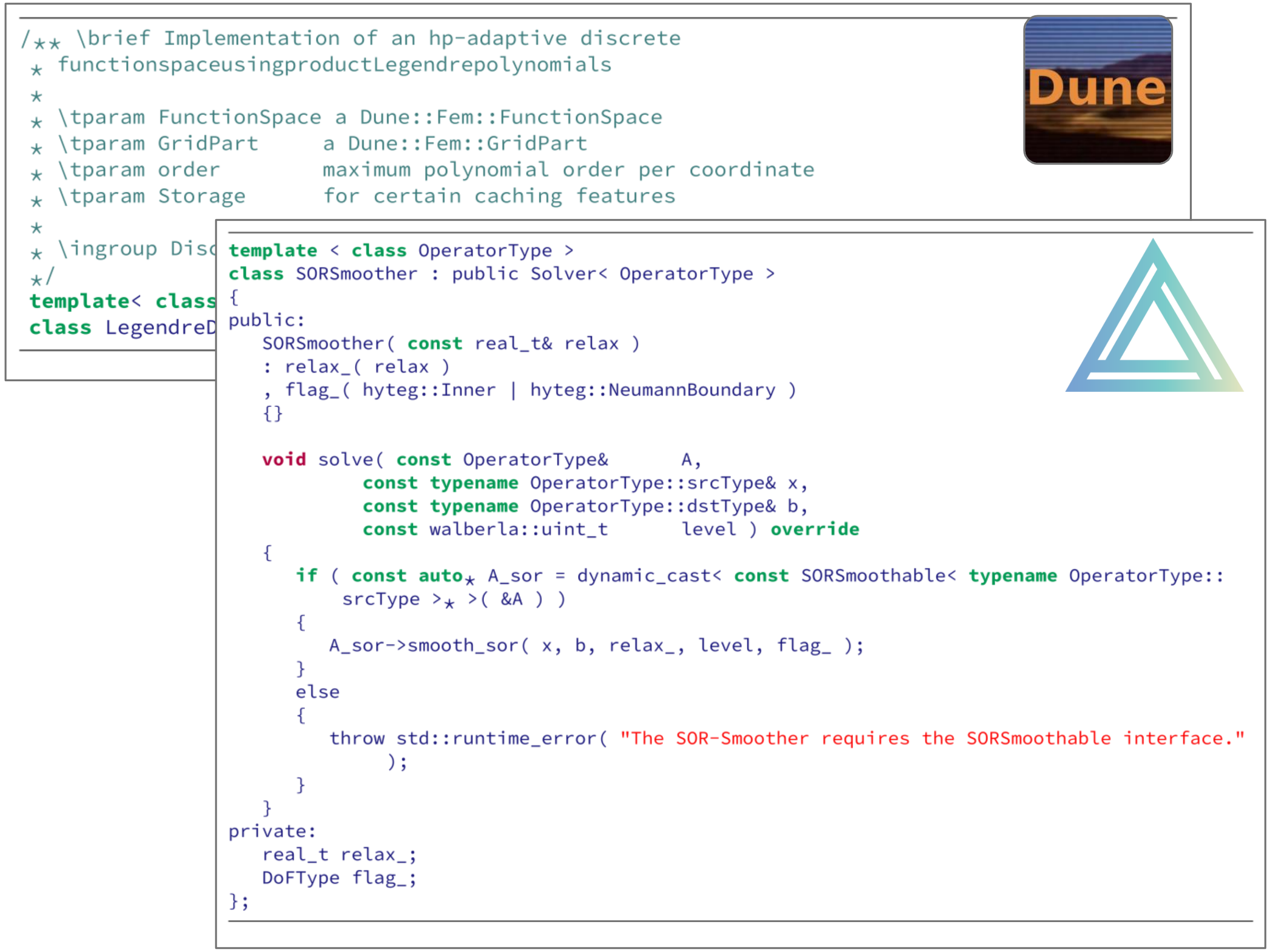
(for compile-time configurable systems)

Compile-Time Configurability

Implementation



```
#include <iostream>
using namespace std;

template <typename AlgoTy>
struct ConfigTraits {
  static void featureA(AlgoTy &Algo) {
    Algo.runFeatureA();
  }

  static void featureB(AlgoTy &Algo, size_t n) {
    Algo.runFeatureB(n);
  }
};

template <typename AlgoTy>
struct GenericImpl {
  void run(size_t n){
    ConfigTraits<AlgoTy>::featureA(_alg);
    ConfigTraits<AlgoTy>::featureB(_alg, n);
  }

  private: AlgoTy _alg{};
};
```



```
/** \brief Implementation of an hp-adaptive discrete
 * functionspaceusingproductLegendrepolynomials
 *
 * \tparam FunctionSpace a Dune::Fem::FunctionSpace
 * \tparam GridPart      a Dune::Fem::GridPart
 * \tparam order         maximum polynomial order per coordinate
 * \tparam Storage       for certain caching features
 *
 * \ingroup Disc
 */
template< class
class LegendreD
```

```
template < class OperatorType >
class SORSmoother : public Solver< OperatorType >
{
public:
   SORSmoother( const real_t& relax )
   : relax_( relax )
   , flag_( hyteg::Inner | hyteg::NeumannBoundary )
   {}

   void solve( const OperatorType&       A,
               const typename OperatorType::srcType& x,
               const typename OperatorType::dstType& b,
               const walberla::uint_t        level ) override
   {
      if ( const auto* A_sor = dynamic_cast< const SORSmoothable< typename OperatorType::
            srcType >* >( &A ) )
      {
         A_sor->smooth_sor( x, b, relax_, level, flag_ );
      }
      else
      {
         throw std::runtime_error( "The SOR-Smoother requires the SORSmoothable interface."
               );
      }
   }
private:
   real_t relax_;
   DoFType flag_;
};
```

Configurable-Software Systems

Subject Systems

- Synthetic Examples

- Handcrafted
- Tailored to different template-implementation techniques

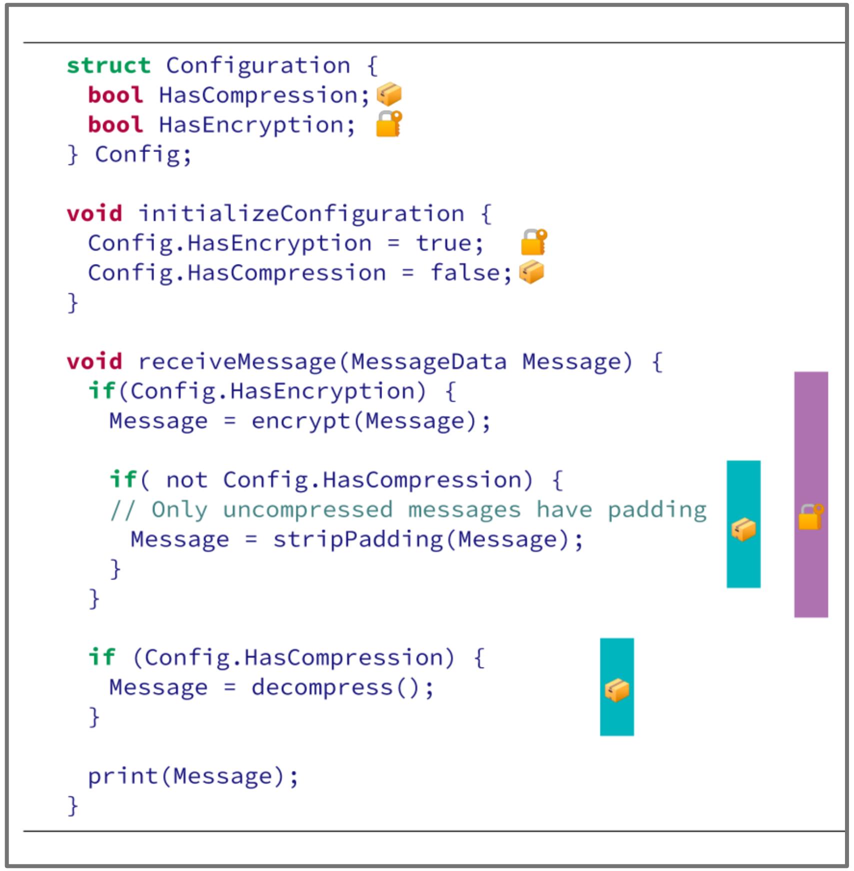
- Real-World Systems

- Real application
- Contact with domain experts

Configuration-Aware Performance Analysis

Overview

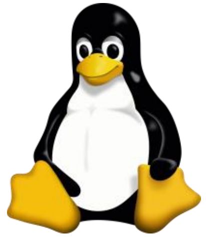
- Identifies feature regions in the source code
- Embeds information into compiled binary
- Combine with performance profilers



```
struct Configuration {
  bool HasCompression;📦
  bool HasEncryption; 🔒
} Config;

void initializeConfiguration {
  Config.HasEncryption = true;  🔒
  Config.HasCompression = false;📦
}

void receiveMessage(MessageData Message) {
  if(Config.HasEncryption) {
    Message = encrypt(Message);

    if( not Config.HasCompression) {
    // Only uncompressed messages have padding
      Message = stripPadding(Message);
    }
  }

  if (Config.HasCompression) {
    Message = decompress();
  }

  print(Message);
}
```

LLVM XRay

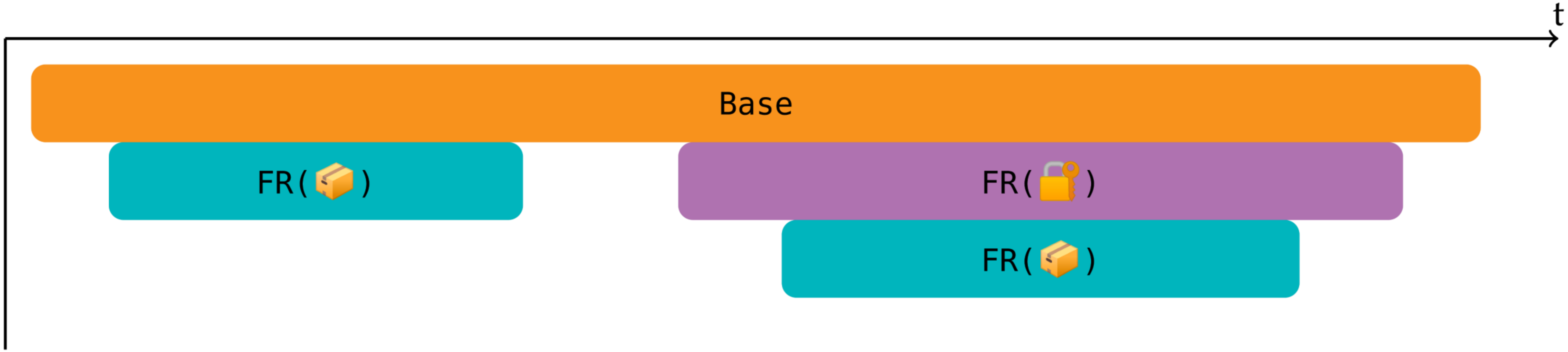
eBPF Framework



PIMTracer

Configuration-Aware Performance Analysis

Performance Influence Model (PIM)



Configuration-Aware Performance Profiling

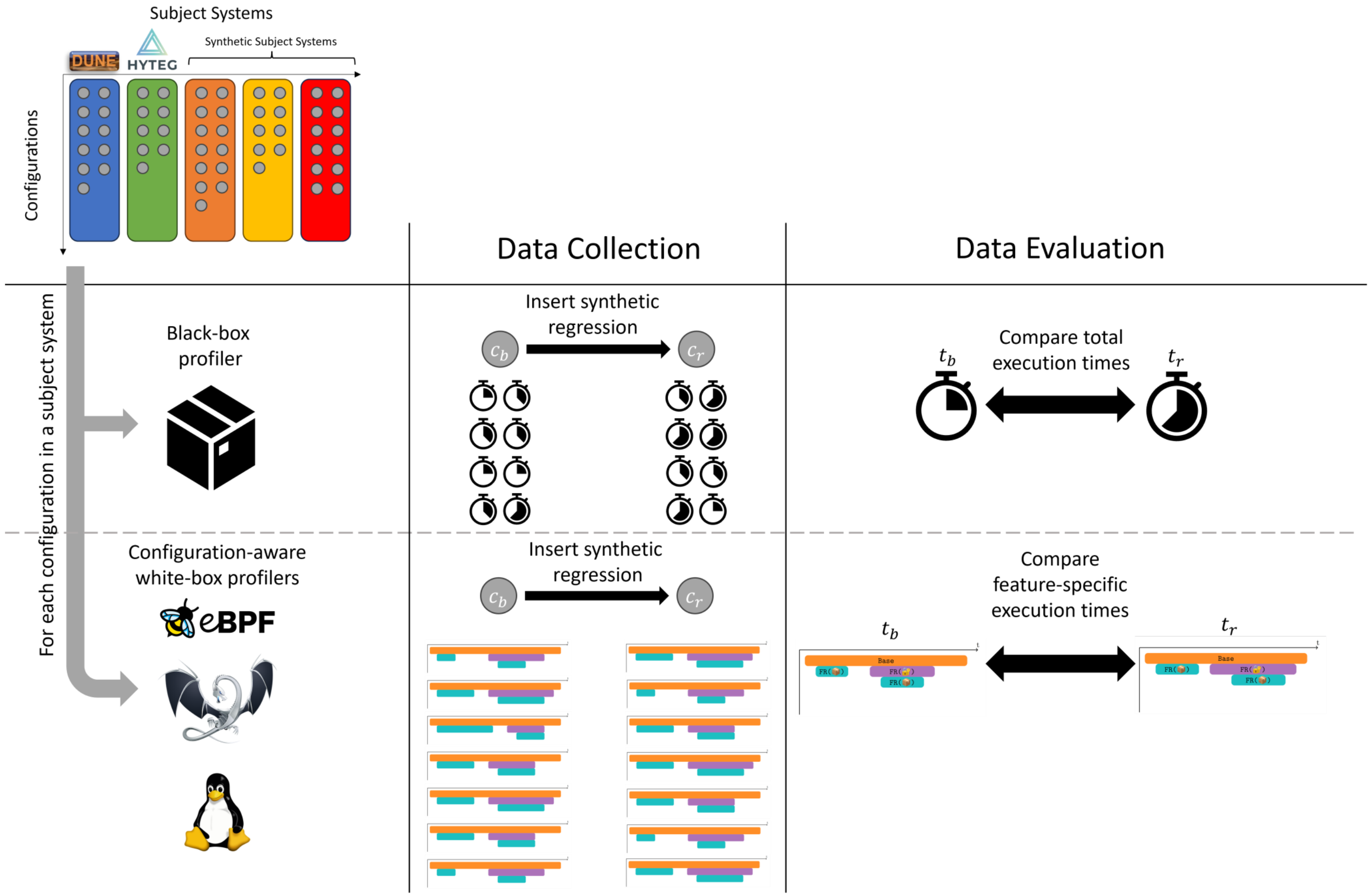
Experiments

Properties of Interest:

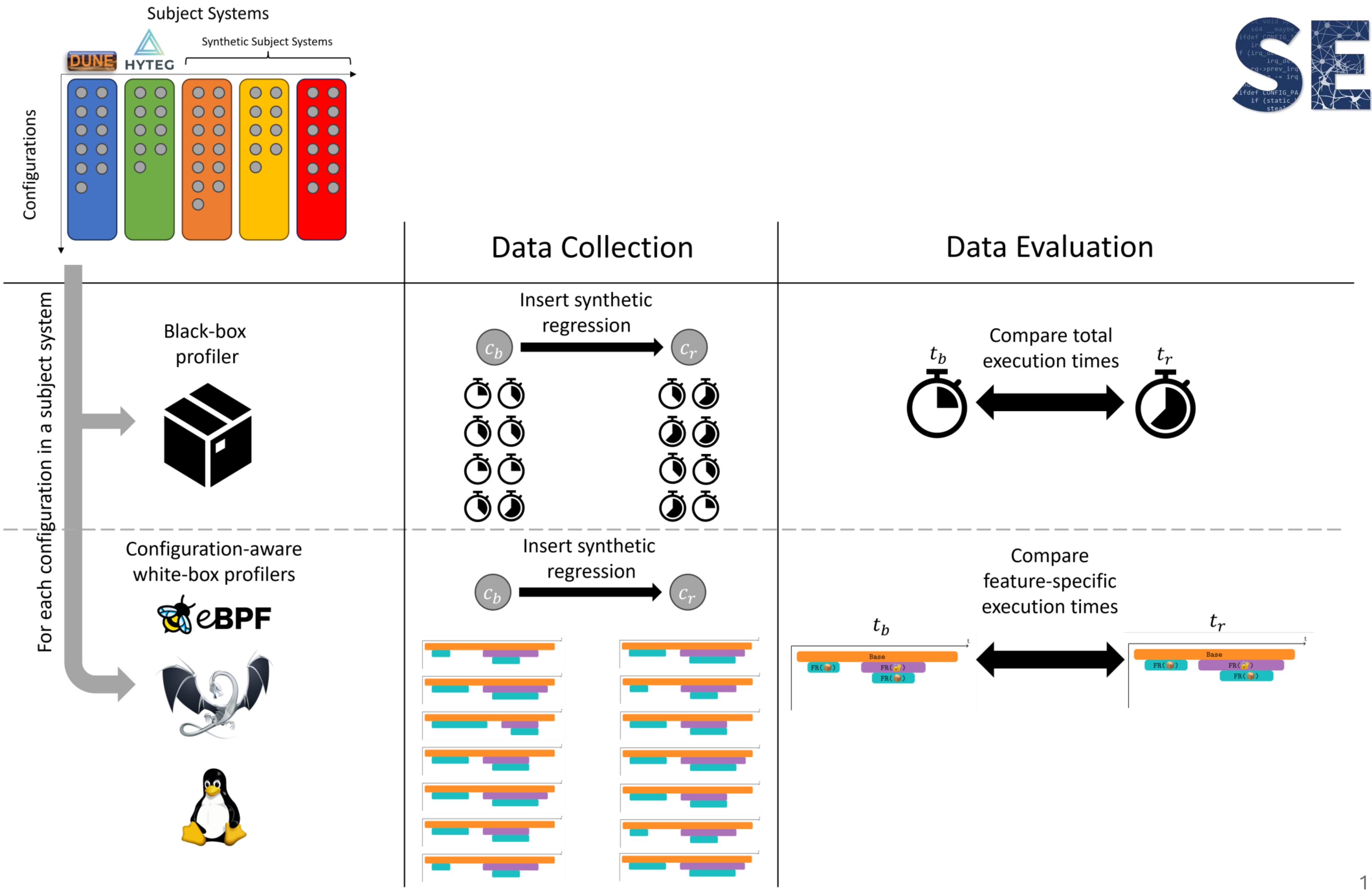
1. Sensitivity
2. Precision & Recall
3. Accuracy

Configuration-Aware Performance Profiling

Experiment Pipeline



Subject Systems
DUNE
HYTEG
Synthetic Subject Systems
Configurations
For each configuration in a subject system
Data Collection
Data Evaluation
Black-box profiler
Insert synthetic regression
c_b
c_r
Compare total execution times
t_b
t_r
Configuration-aware white-box profilers
eBPF
Insert synthetic regression
c_b
c_r
Compare feature-specific execution times
t_b
t_r



Subject Systems

Synthetic Subject Systems

Configurations

Data Collection

Data Evaluation

For each configuration in a subject system

Black-box profiler

Insert synthetic regression

c_b c_r

Compare total execution times

t_b t_r

Configuration-aware white-box profilers

Insert synthetic regression

c_b c_r

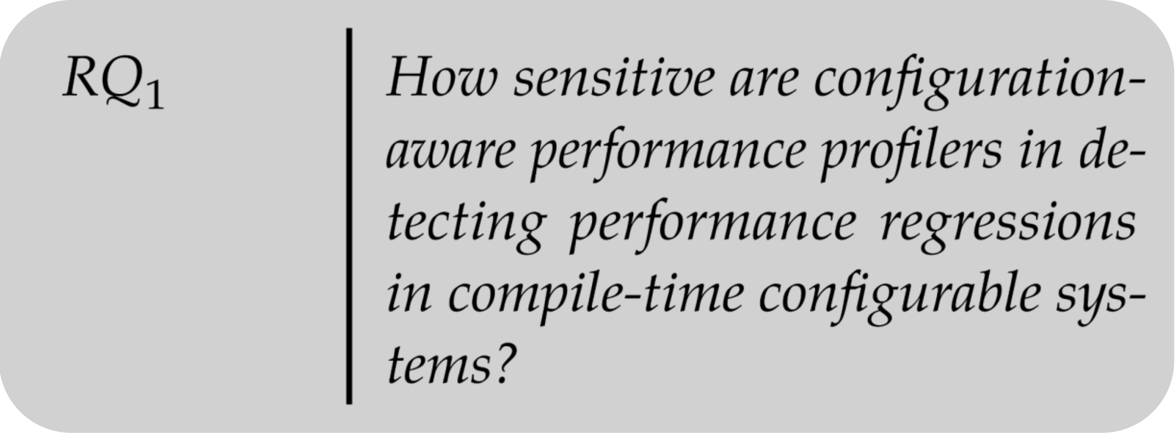
Compare feature-specific execution times

t_b t_r

RQ_1 - Sensitivity

Overview

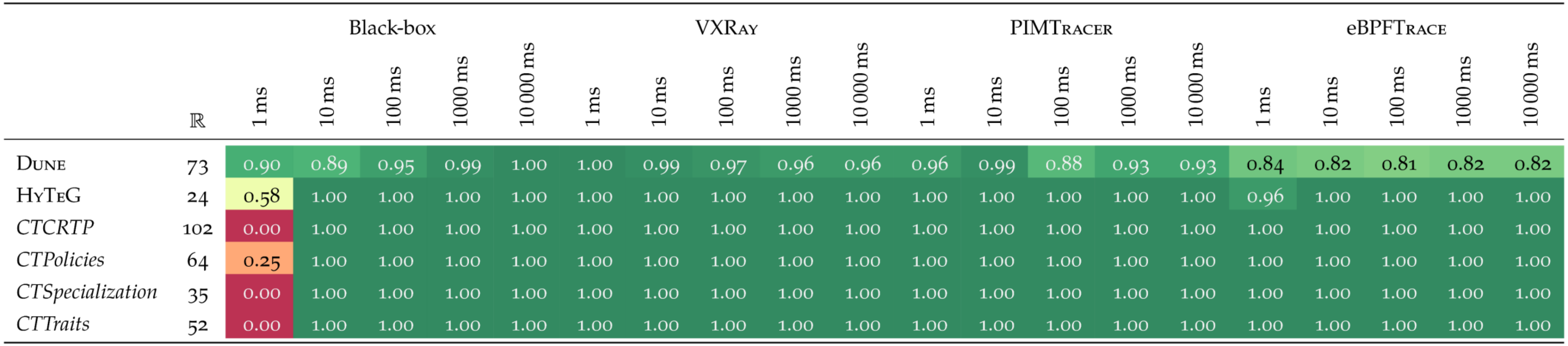
- Regression selection:
    - Introduce regressions for single features of varying severity (10 s, 1 s, 100 ms, 10 ms, 1ms)
- Data Collection:
    - Black-box: Total time for execution
    - White-box: Total time spent in feature code only
- Data Evaluation:
    - Independent t-test between base and regressed times



RQ_1 | *How sensitive are configuration-aware performance profilers in detecting performance regressions in compile-time configurable systems?*

RQ_1 - Sensitivity

Results

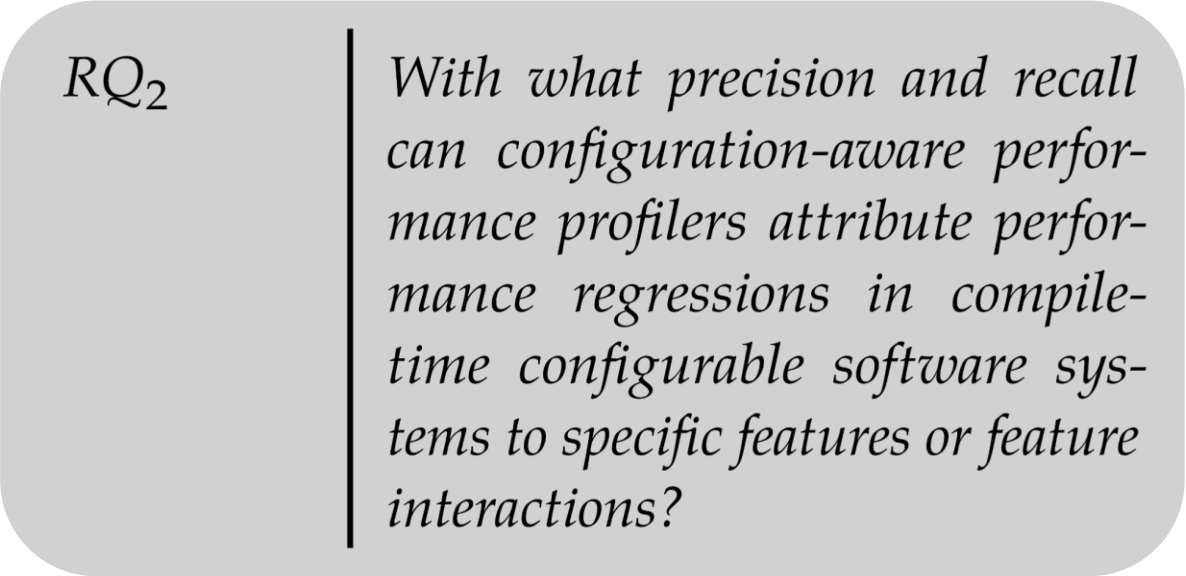


	ℝ	Black-box					VXRay					PIMTracer					eBPFTrace				
		1 ms	10 ms	100 ms	1000 ms	10 000 ms	1 ms	10 ms	100 ms	1000 ms	10 000 ms	1 ms	10 ms	100 ms	1000 ms	10 000 ms	1 ms	10 ms	100 ms	1000 ms	10 000 ms
Dune	73	0.90	0.89	0.95	0.99	1.00	1.00	0.99	0.97	0.96	0.96	0.96	0.99	0.88	0.93	0.93	0.84	0.82	0.81	0.82	0.82
HyTeG	24	0.58	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	0.96	1.00	1.00	1.00	1.00
CTCRTP	102	0.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
CTPolicies	64	0.25	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
CTSpecialization	35	0.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
CTTraits	52	0.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00

RQ_2 – Precision & Recall

Overview

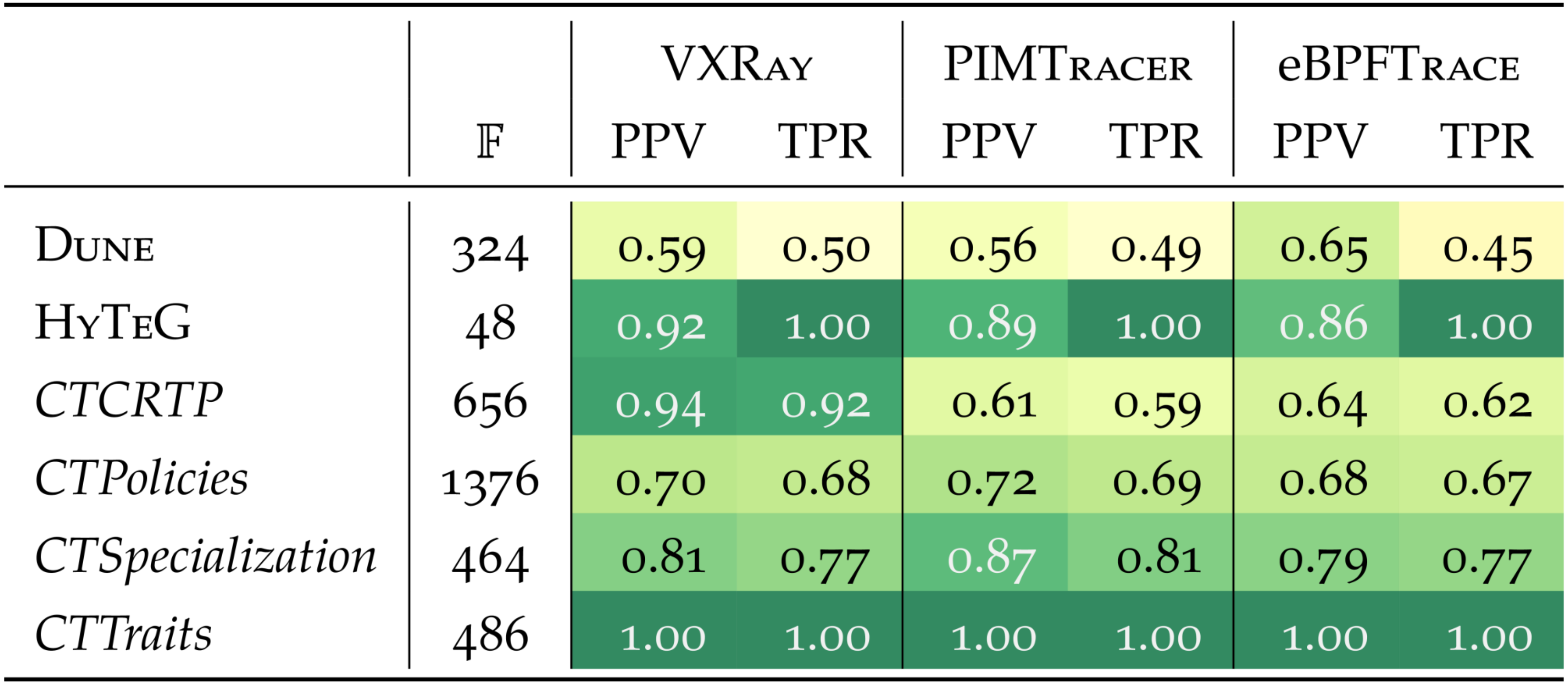
- Regression selection:
  - Create regressions affecting one or multiple features
- Data Collection:
  - Compare execution times for each individual feature
- Data Evaluation:
  - Compare based and regressed variants
  - Identify TP/FP/TN/FN
  - Calculate Precision and Recall



RQ_2 | *With what precision and recall can configuration-aware performance profilers attribute performance regressions in compile-time configurable software systems to specific features or feature interactions?*

RQ_2 – Precision & Recall

Results

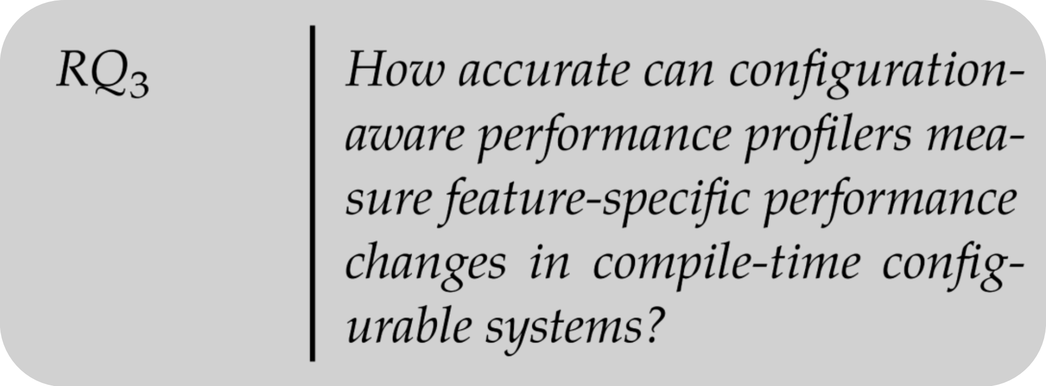


	$\mathbb{F}$	VXRay		PIMTracer		eBPFTrace	
		PPV	TPR	PPV	TPR	PPV	TPR
Dune	324	0.59	0.50	0.56	0.49	0.65	0.45
HyTeG	48	0.92	1.00	0.89	1.00	0.86	1.00
CTCRTP	656	0.94	0.92	0.61	0.59	0.64	0.62
CTPolicies	1376	0.70	0.68	0.72	0.69	0.68	0.67
CTSpecialization	464	0.81	0.77	0.87	0.81	0.79	0.77
CTTraits	486	1.00	1.00	1.00	1.00	1.00	1.00

RQ_3 – Accuracy

Results

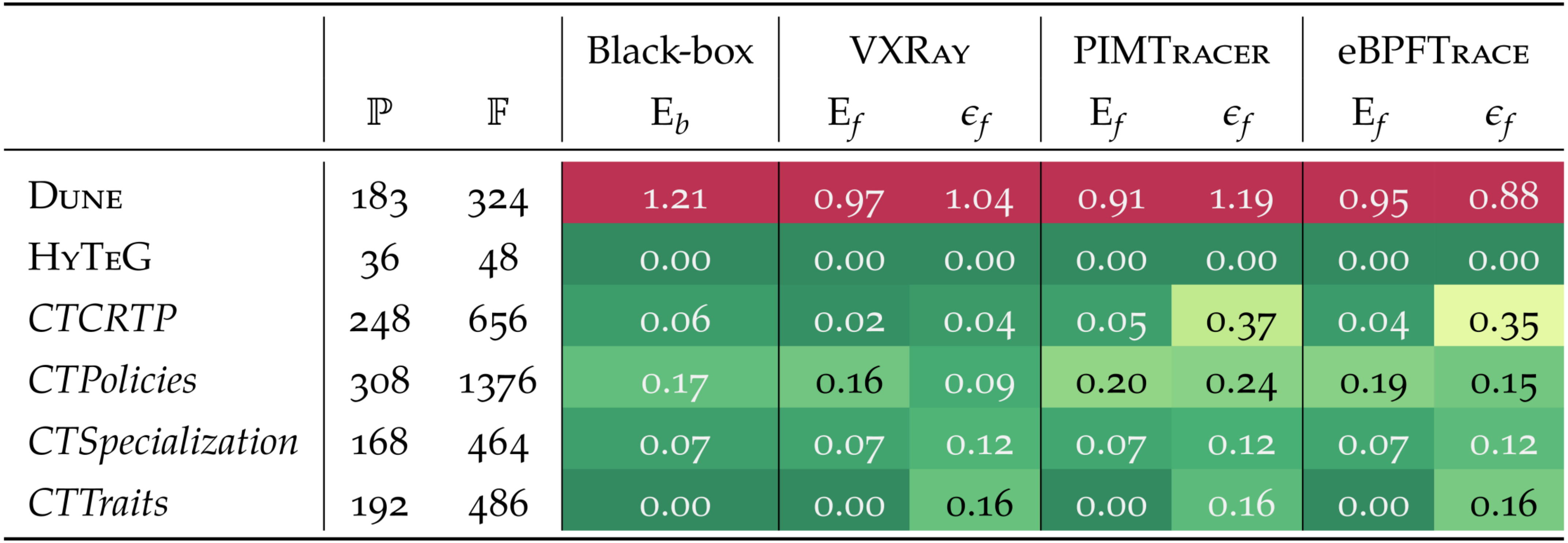
- Regression selection:
  - Create regressions affecting one or multiple features (Same as RQ2)
- Data Collection:
  - Black-box: Total time difference for execution
  - White-box:
    - Total time difference across all feature code
    - Time difference for each individual feature
- Data Evaluation:
  - Overall error (Black-box & White-box)
  - Feature-specific error (White-box)



RQ_3 | *How accurate can configuration-aware performance profilers measure feature-specific performance changes in compile-time configurable systems?*

RQ_3 – Accuracy

Results



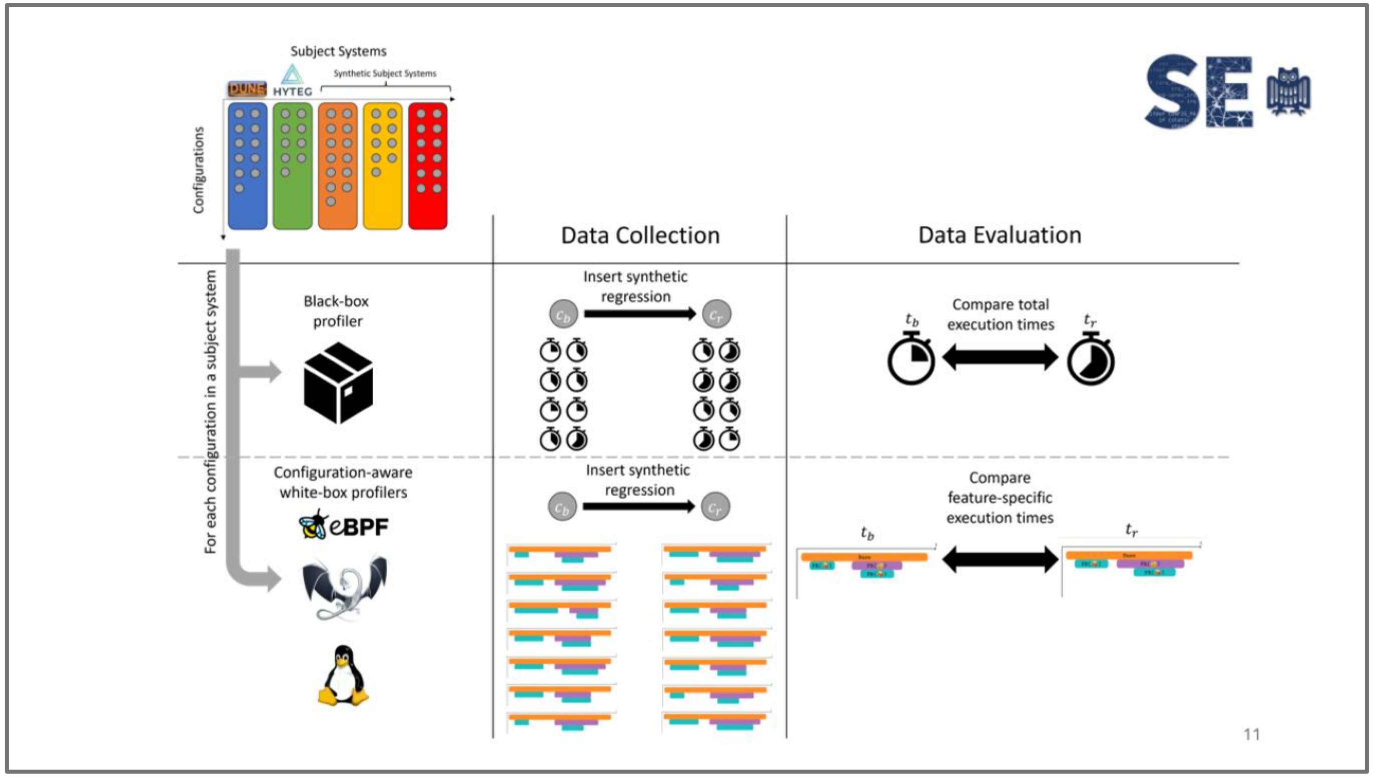
			Black-box	VXRay		PIMTracer		eBPFTrace	
	$\mathbb{P}$	$\mathbb{F}$	E_b	E_f	ϵ_f	E_f	ϵ_f	E_f	ϵ_f
Dune	183	324	1.21	0.97	1.04	0.91	1.19	0.95	0.88
HyTeG	36	48	0.00	0.00	0.00	0.00	0.00	0.00	0.00
CTCRTP	248	656	0.06	0.02	0.04	0.05	0.37	0.04	0.35
CTPolicies	308	1376	0.17	0.16	0.09	0.20	0.24	0.19	0.15
CTSpecialization	168	464	0.07	0.07	0.12	0.07	0.12	0.07	0.12
CTTraits	192	486	0.00	0.00	0.16	0.00	0.16	0.00	0.16

Summary

Results & Outlook

- White-box profilers are…
  - …as capable as black-box profilers
  - …**and** can provide us with more feature-specific information

- Results vary by…
  - Implementation Pattern
  - Profiling technology



Subject Systems

Synthetic Subject Systems

Data Collection

Data Evaluation

For each configuration in a subject system

Black-box profiler

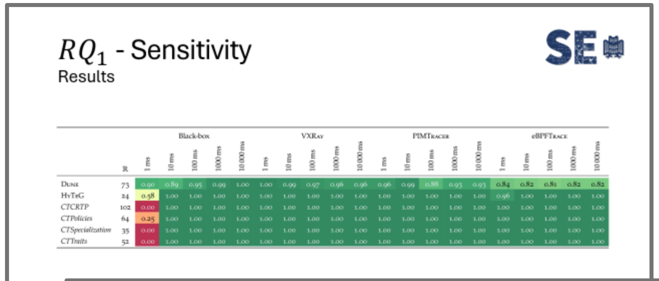
Insert synthetic regression

Compare total execution times

Configuration-aware white-box profilers

Insert synthetic regression

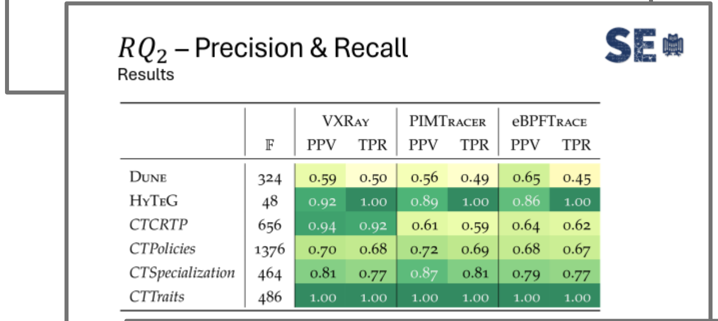
Compare feature-specific execution times



RQ_1 - Sensitivity

Results

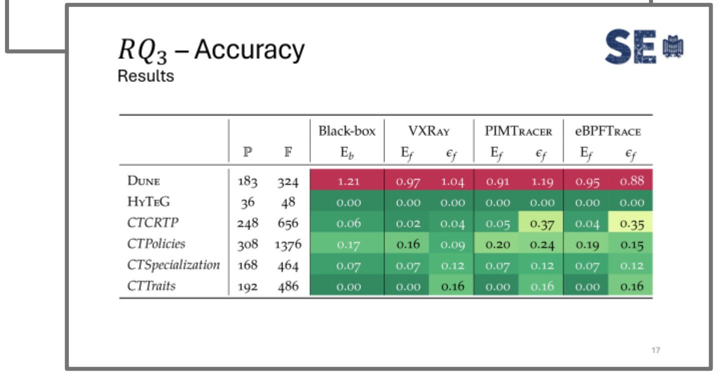
	R	Black-box 1 ms	10 ms	100 ms	1000 ms	10 000 ms	VXRay 1 ms	10 ms	100 ms	1000 ms	10 000 ms	PIMTracer 1 ms	10 ms	100 ms	1000 ms	10 000 ms	eBPFTrace 1 ms	10 ms	100 ms	1000 ms	10 000 ms
Dune	73	0.90	0.89	0.95	0.99	1.00	1.00	0.99	0.97	0.96	0.96	0.96	0.99	0.88	0.93	0.93	0.84	0.82	0.81	0.82	0.82
HyTeG	24	0.38	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	0.96	1.00	1.00	1.00	1.00
CTCRTP	102	0.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
CTPolicies	64	0.25	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
CTSpecialization	35	0.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
CTTraits	52	0.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00



RQ_2 – Precision & Recall

Results

	F	VXRay PPV	VXRay TPR	PIMTracer PPV	PIMTracer TPR	eBPFTrace PPV	eBPFTrace TPR
Dune	324	0.59	0.50	0.56	0.49	0.65	0.45
HyTeG	48	0.92	1.00	0.89	1.00	0.86	1.00
CTCRTP	656	0.94	0.92	0.61	0.59	0.64	0.62
CTPolicies	1376	0.70	0.68	0.72	0.69	0.68	0.67
CTSpecialization	464	0.81	0.77	0.87	0.81	0.79	0.77
CTTraits	486	1.00	1.00	1.00	1.00	1.00	1.00



RQ_3 – Accuracy

Results

	P	F	Black-box E_b	VXRay E_f	VXRay ϵ_f	PIMTracer E_f	PIMTracer ϵ_f	eBPFTrace E_f	eBPFTrace ϵ_f
Dune	183	324	1.21	0.97	1.04	0.91	1.19	0.95	0.88
HyTeG	36	48	0.00	0.00	0.00	0.00	0.00	0.00	0.00
CTCRTP	248	656	0.06	0.02	0.04	0.05	0.37	0.04	0.35
CTPolicies	308	1376	0.17	0.16	0.09	0.20	0.24	0.19	0.15
CTSpecialization	168	464	0.07	0.07	0.12	0.07	0.12	0.07	0.12
CTTraits	192	486	0.00	0.00	0.16	0.00	0.16	0.00	0.16

Backup Slides

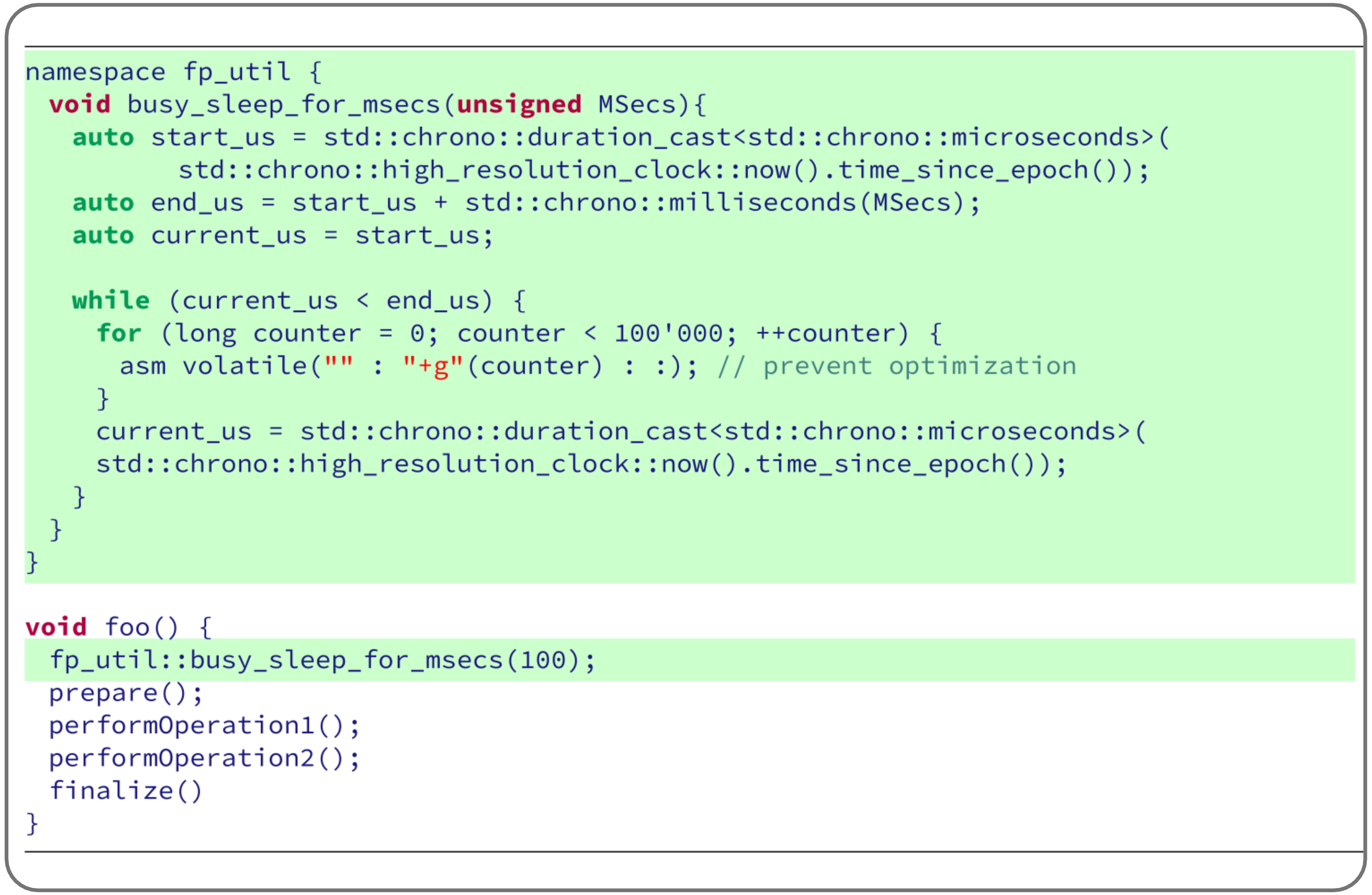
Outlook

- Further Steps:
    - Real-world regressions
        - First results are promising
    - Include more projects

- Extensions and other applications:
    - Information gain of white-box analyses
    - Feature-performance over time
        - Feature flags
        - Feature-dependent performance change points

Experiment Setup

Synthetic Regressions

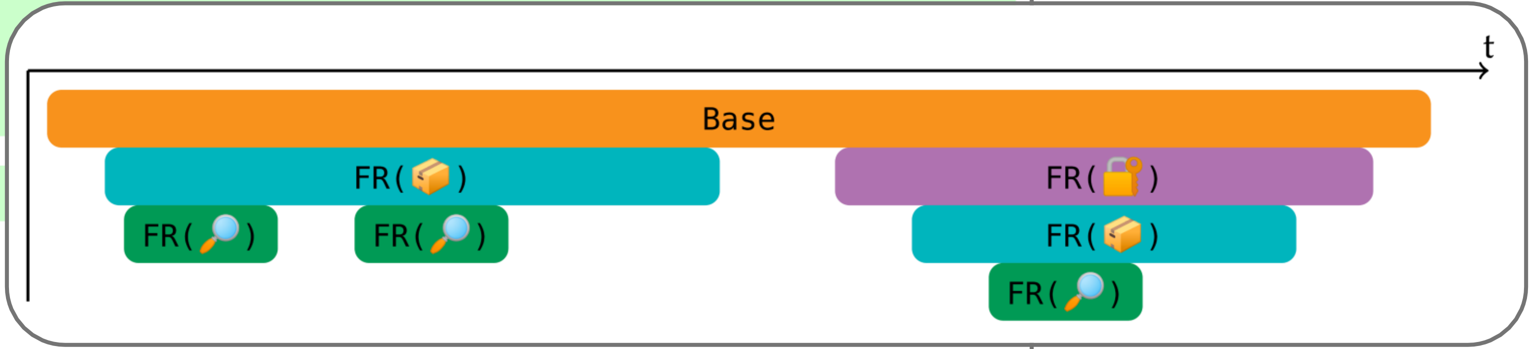


```
namespace fp_util {
  void busy_sleep_for_msecs(unsigned MSecs){
    auto start_us = std::chrono::duration_cast<std::chrono::microseconds>(
             std::chrono::high_resolution_clock::now().time_since_epoch());
    auto end_us = start_us + std::chrono::milliseconds(MSecs);
    auto current_us = start_us;

    while (current_us < end_us) {
      for (long counter = 0; counter < 100'000; ++counter) {
        asm volatile("" : "+g"(counter) : :); // prevent optimization
      }
      current_us = std::chrono::duration_cast<std::chrono::microseconds>(
      std::chrono::high_resolution_clock::now().time_since_epoch());
    }
  }
}

void foo() {
  fp_util::busy_sleep_for_msecs(100);
  prepare();
  performOperation1();
  performOperation2();
  finalize()
}
```

Feature-Specific Ground Truth

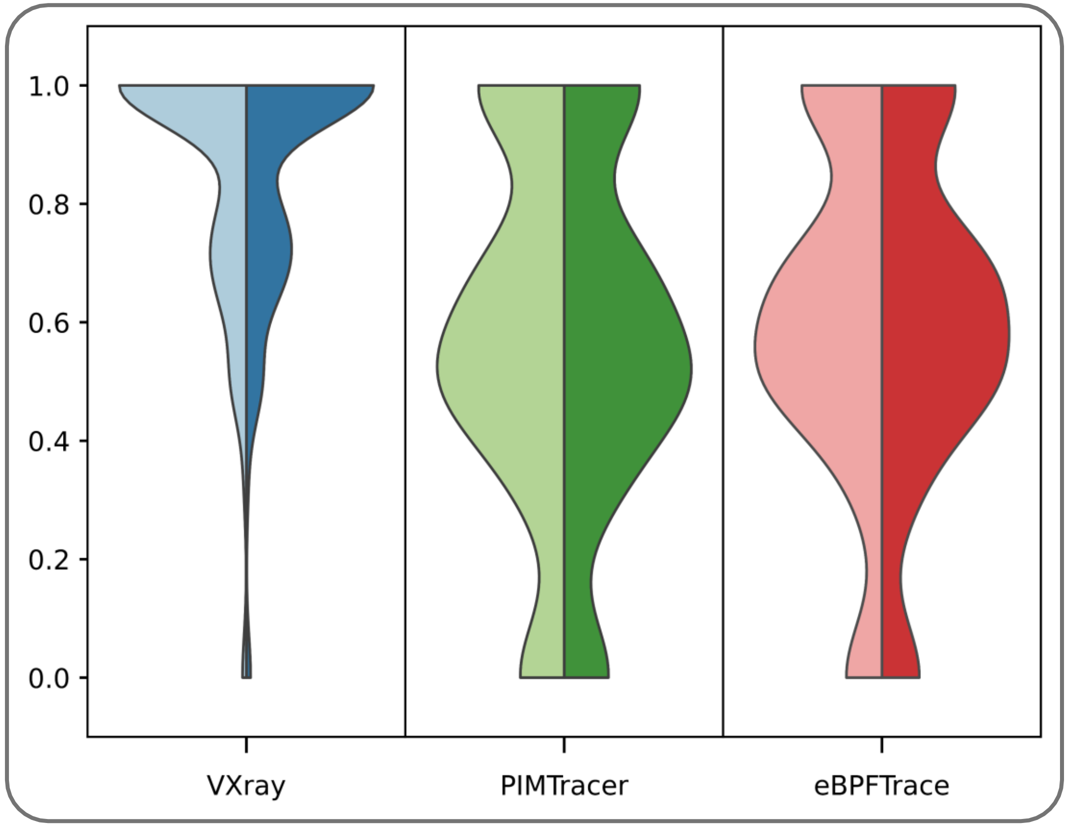
Technical Details

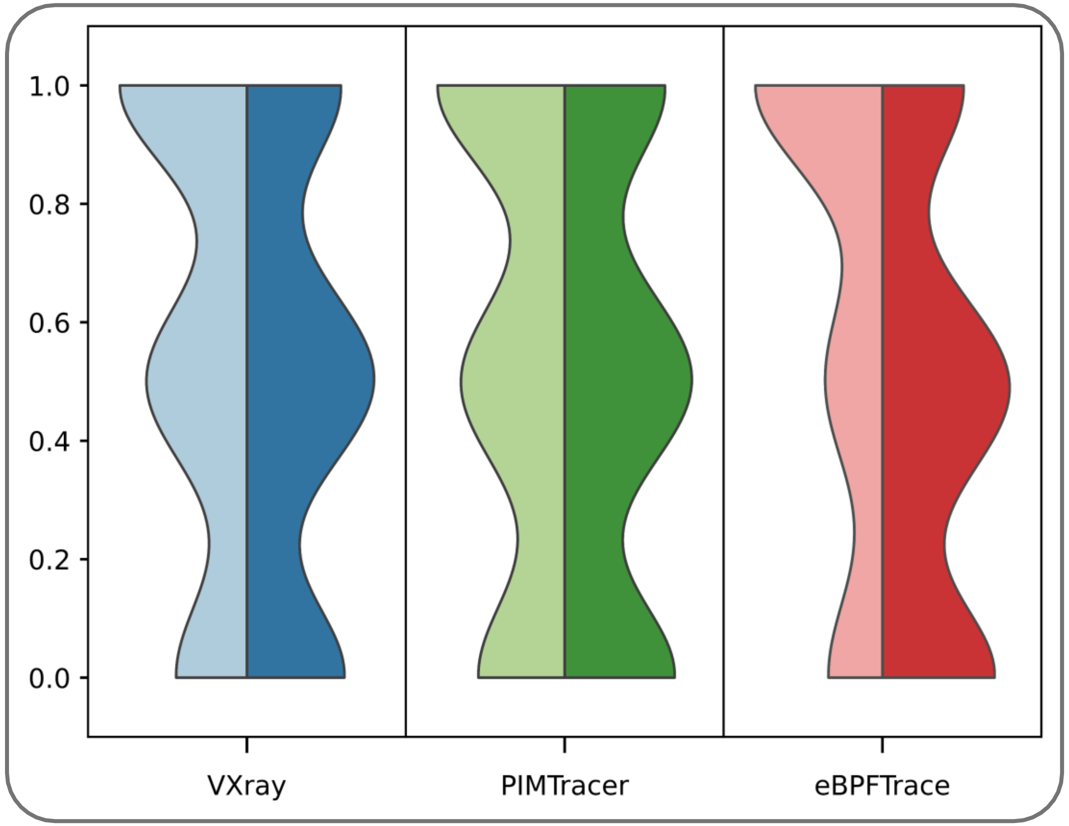
```
namespace fp_util {
  __attribute__((feature_variable("__VARA__DETECT__"))) void detect() {
    long foo = 0;
    asm volatile("" : "+g"(foo) : :);
    foo++;
  }
}

void foo() {
  fp_util::detect();
  prepare();
  performOperation1();
  performOperation2();
  finalize()
}
```



RQ_2 – Precision & Recall

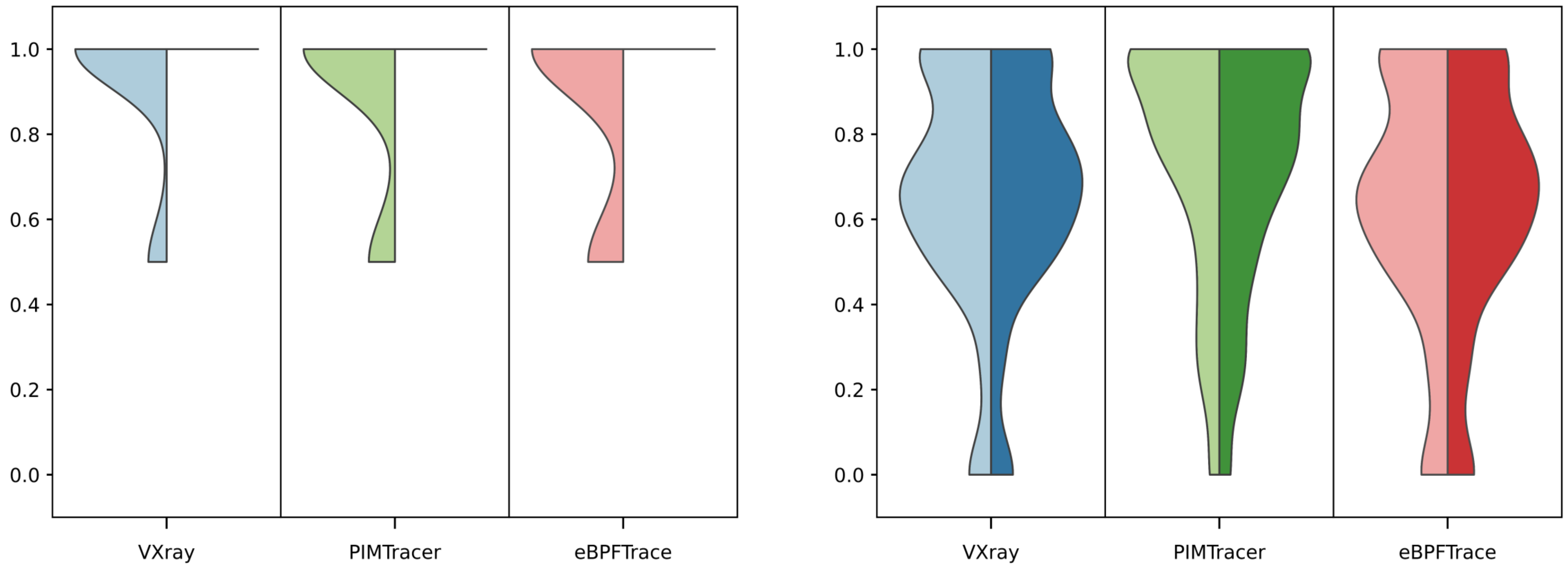
Results





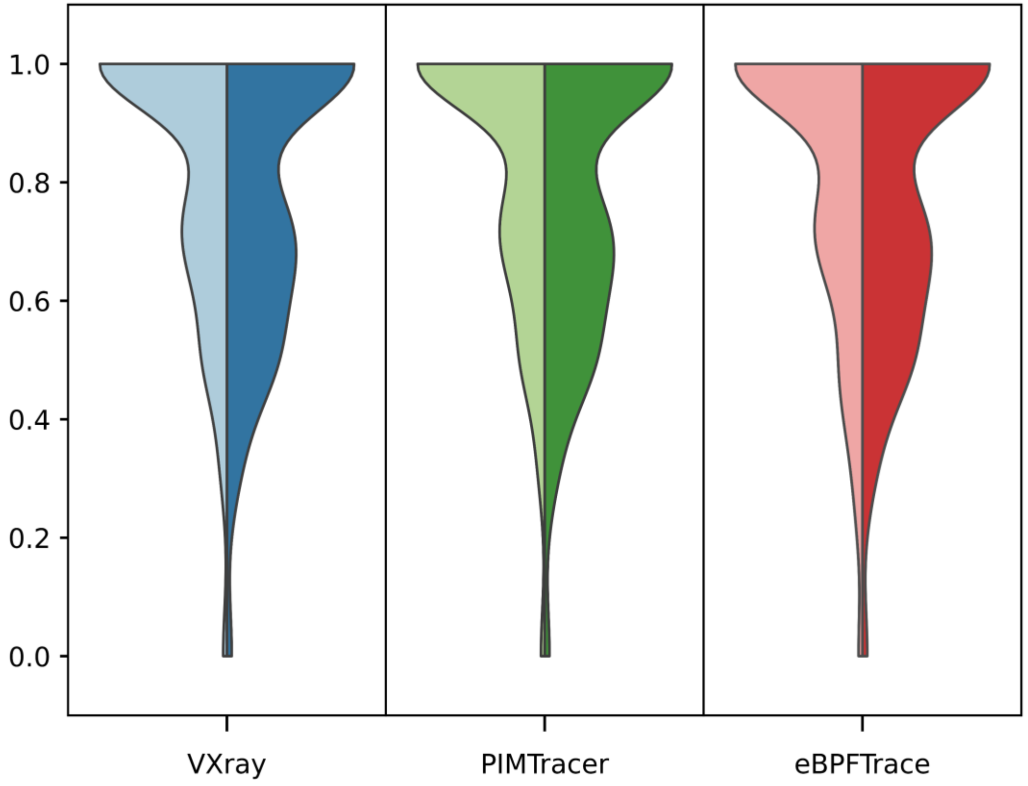
RQ_2 – Precision & Recall

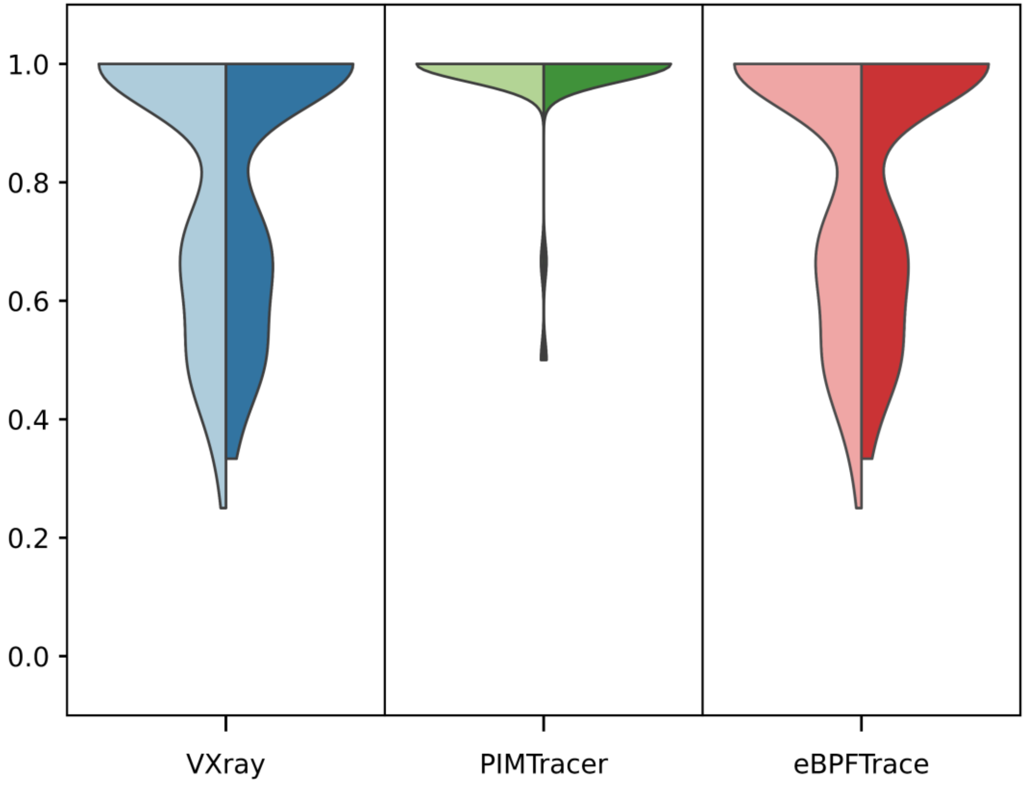
Results



RQ_2 – Precision & Recall

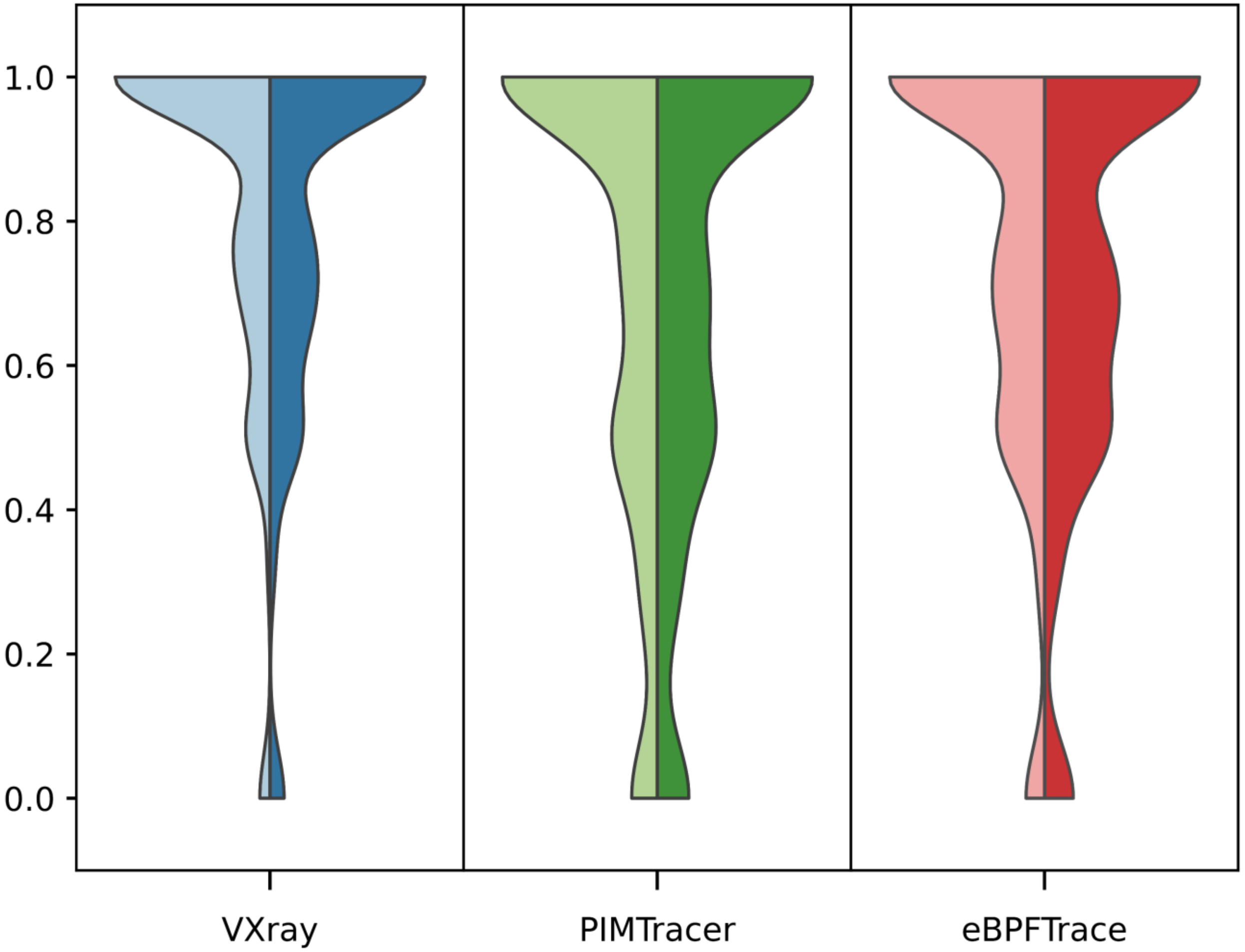
Results





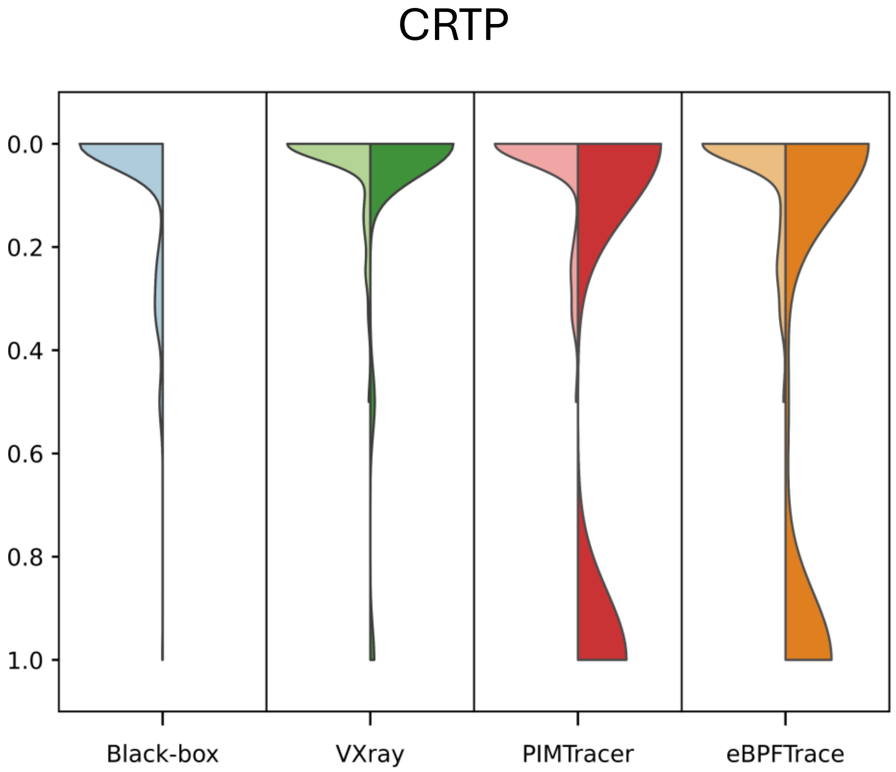
RQ_2 – Precision & Recall

Results

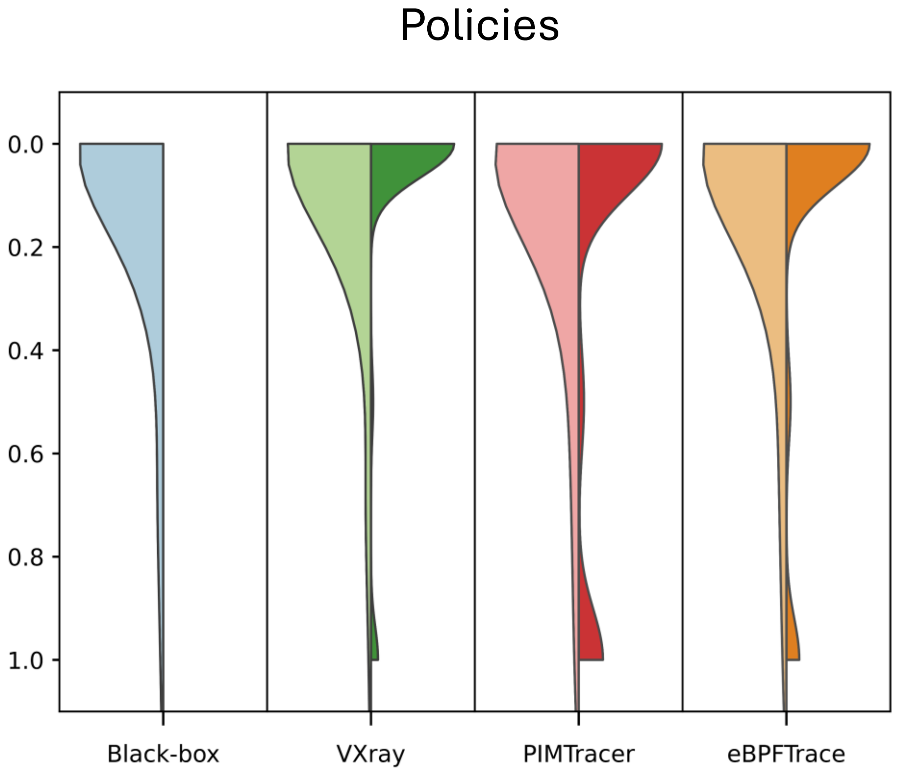


RQ_3 – Accuracy

Results



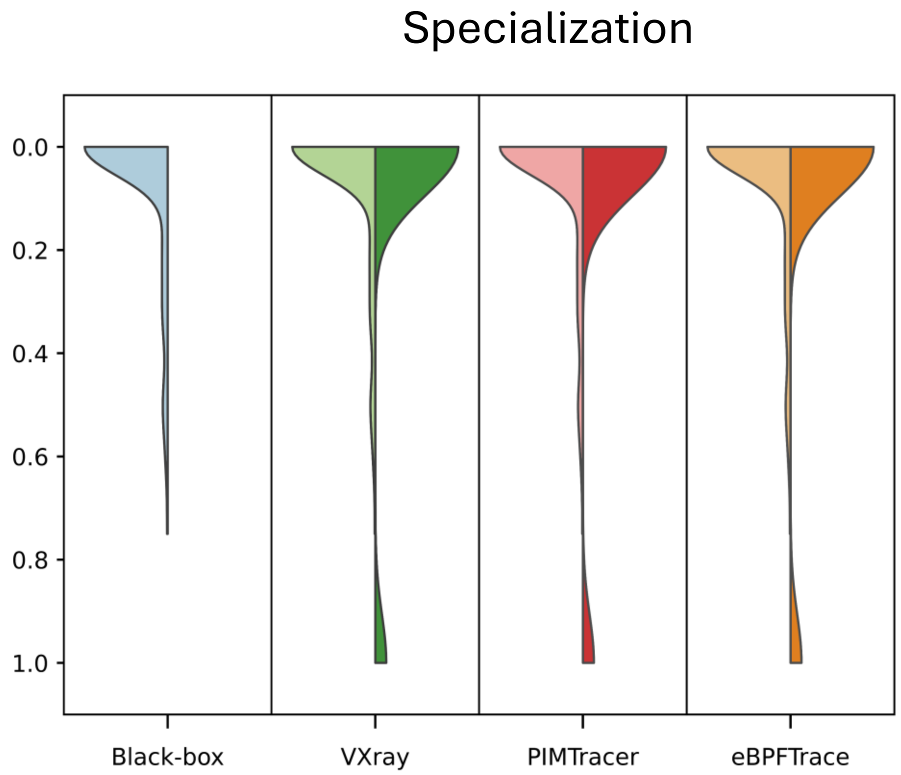
CRTP

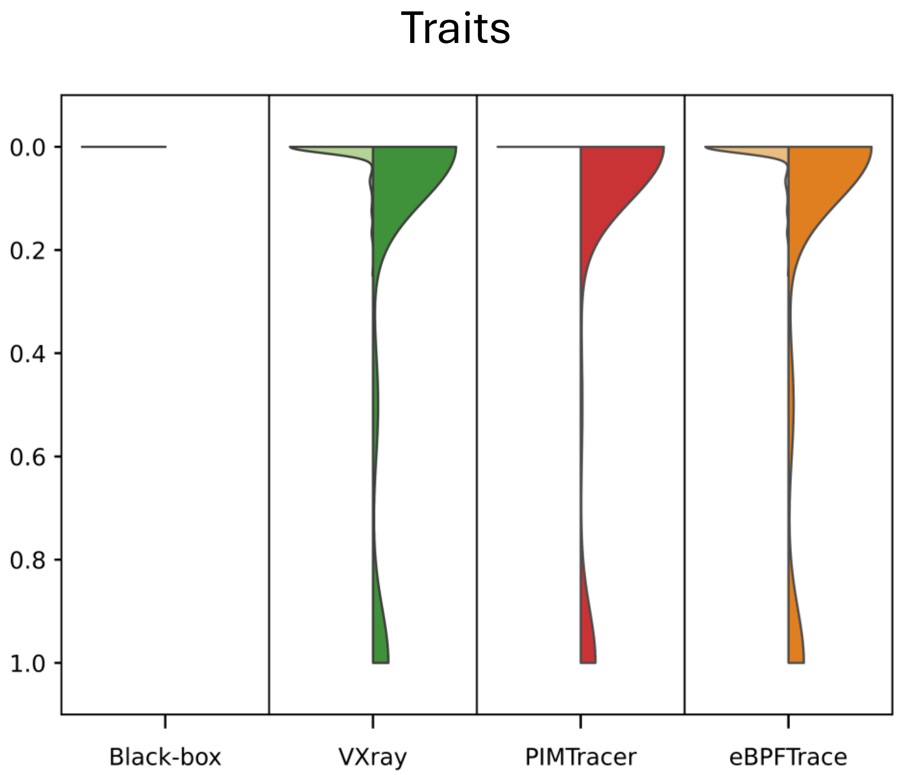


Policies

RQ_3 – Accuracy

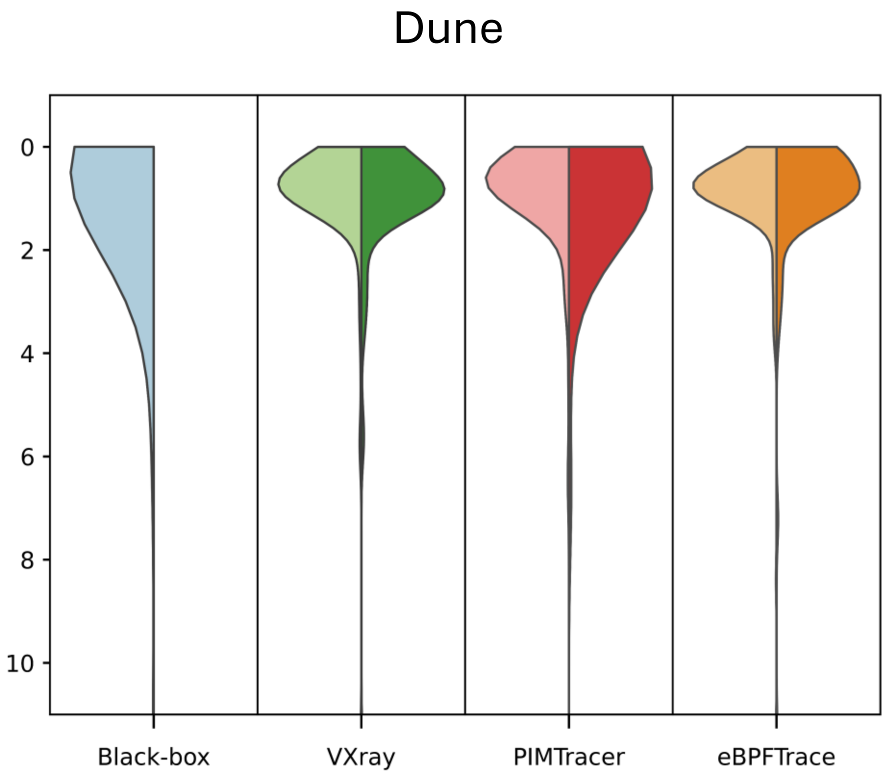
Results





RQ_3 – Accuracy

Results


Dune
Black-box
VXray
PIMTracer
eBPFTrace