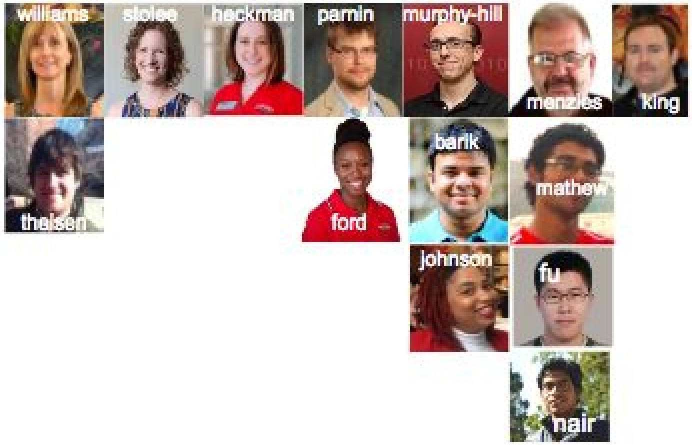
NC STATE UNIVERSITY

software
... if engineering then NC State ...



Using Bad Learners to find Good Configurations

Vivek Nair Tim Menzies
North Carolina State University

Norbert Siegmund
Bauhaus-Universität Weimar

Sven Apel
University of Passau

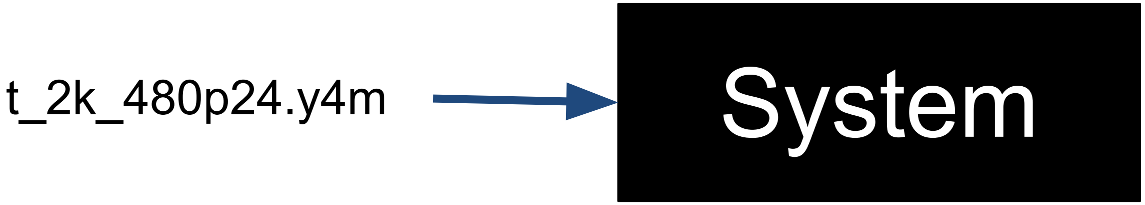
Bauhaus-Universität Weimar

UNIVERSITÄT PASSAU

Configurable Systems and Variability

System

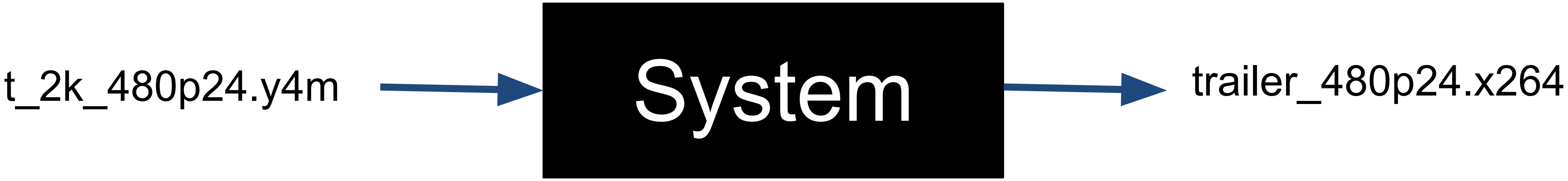
Configurable Systems and Variability



t_2k_480p24.y4m

System

Configurable Systems and Variability

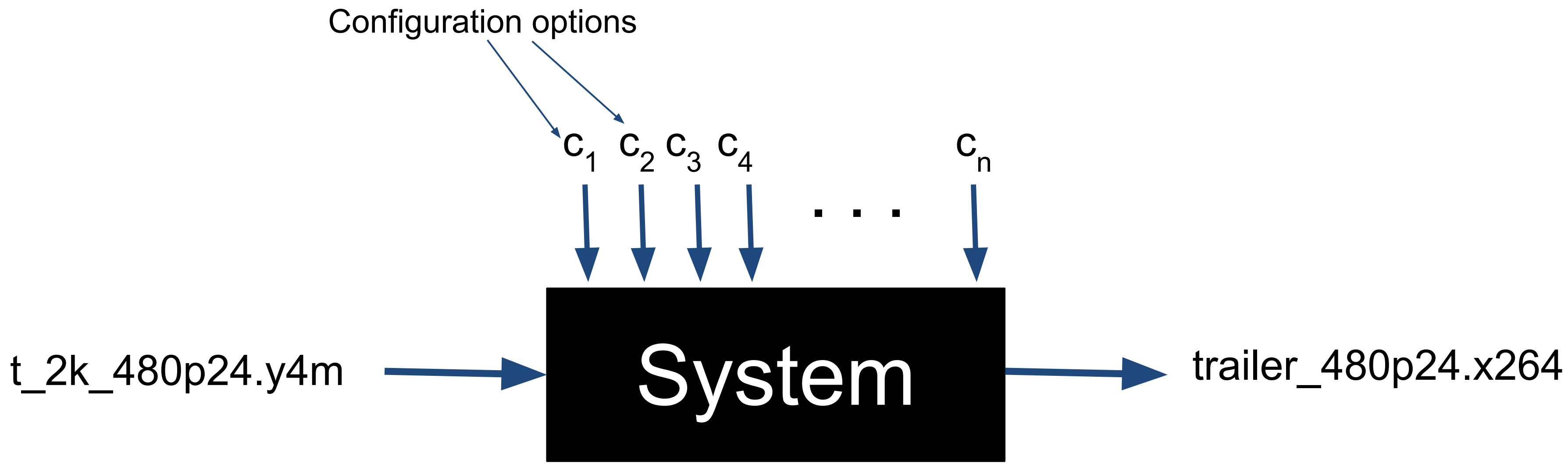


t_2k_480p24.y4m

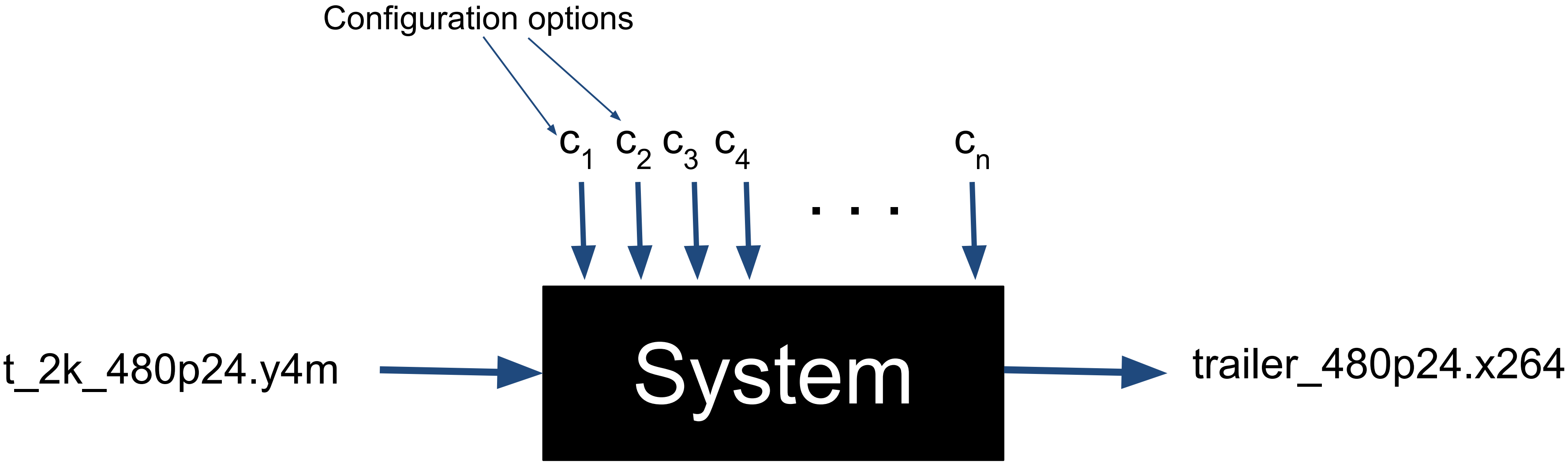
System

trailer_480p24.x264

Configurable Systems and Variability

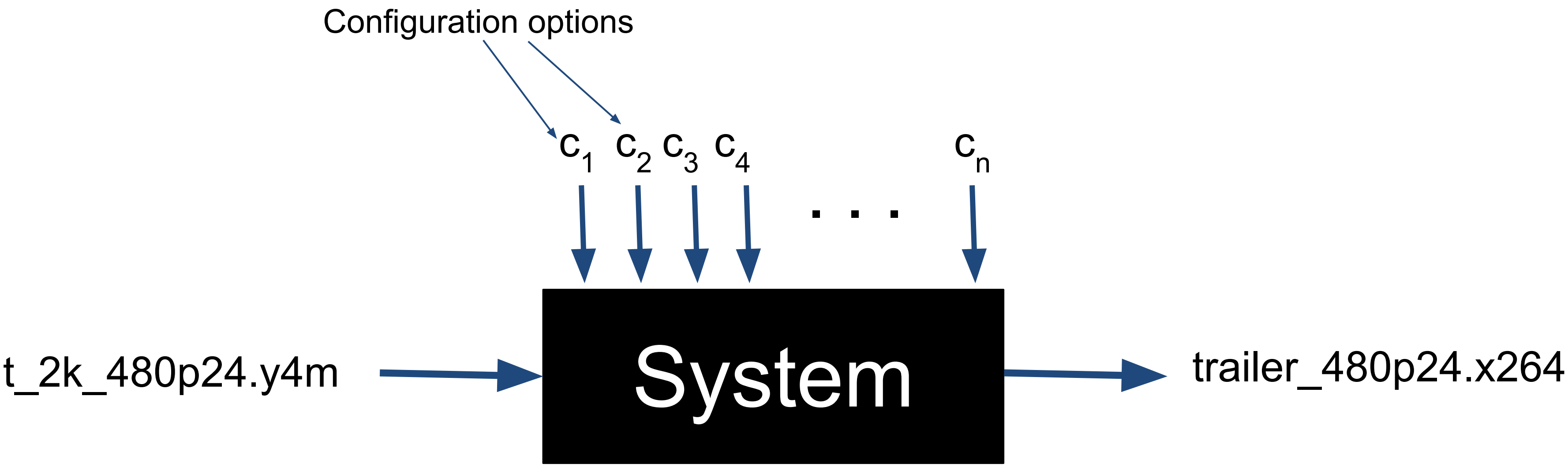


Configurable Systems and Variability



Non-functional behavior: response time, throughput, etc.

Configurable Systems and Variability



Non-functional behavior: response time, throughput, etc.

Objective: Find (near) **optimal configuration** of a system **with minimal effort**

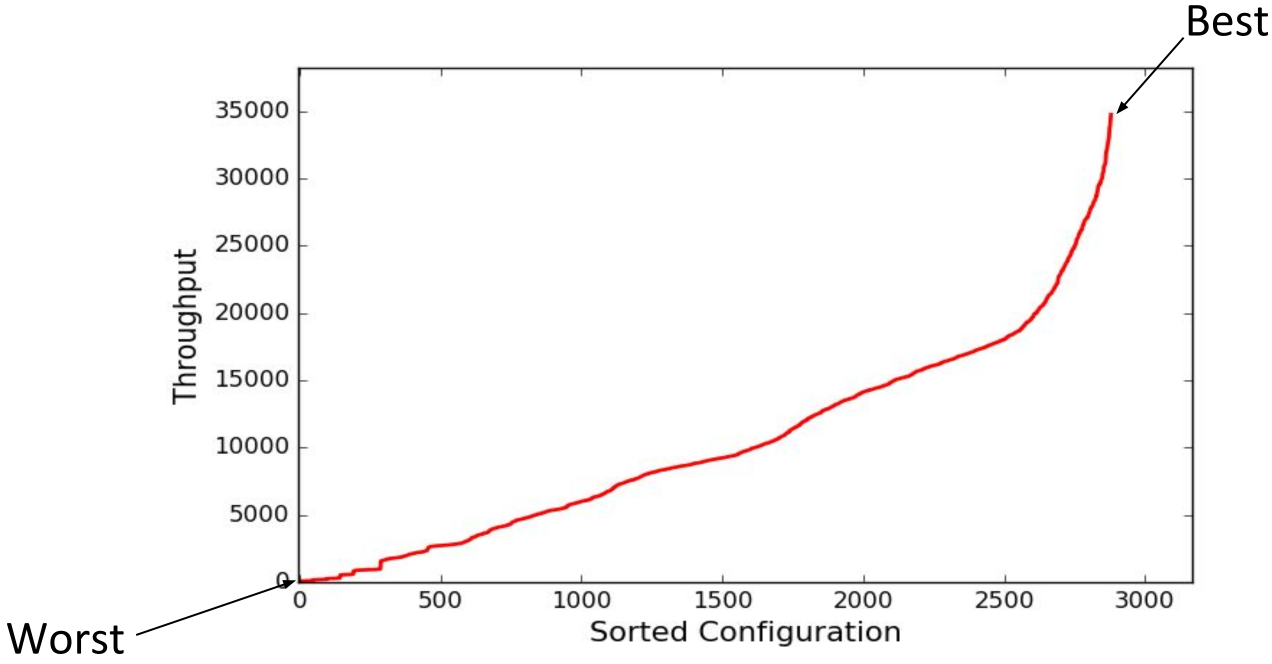
Performance Optimization is Necessary!

System: Apache Storm

Workload: Word Count

Performance: Throughput

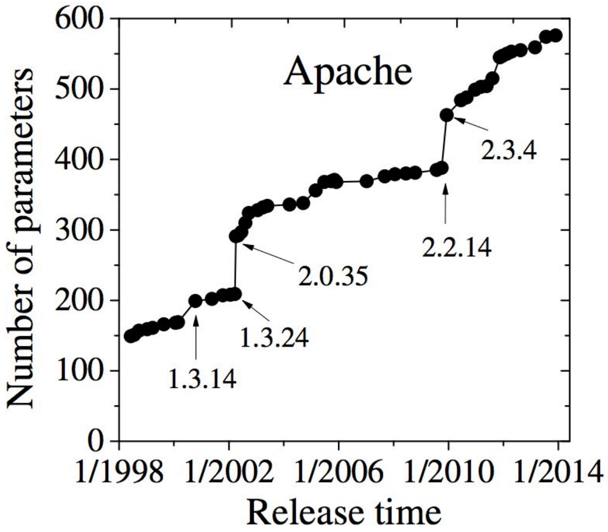
# Configurations: 6

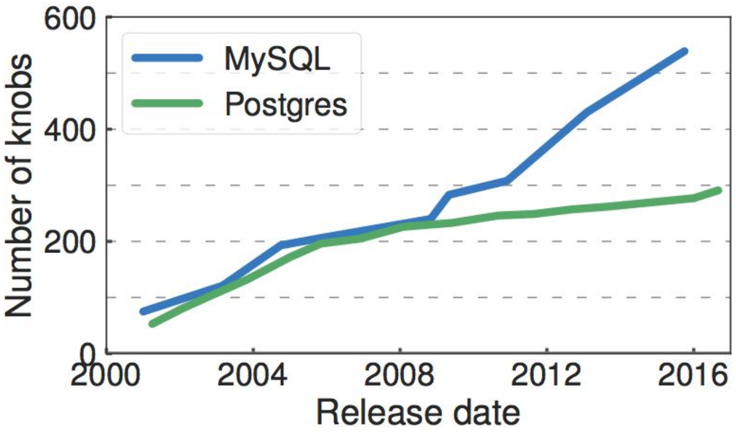


Best configuration is 480 times better than **Worst** configuration

Performance Optimization is getting more Complex!

✓ Necessary





200 new configuration options added to Apache HTTP server between 2010 and 2014

250 new configuration options added to MySQL between 2012 and 2016

[1] Xu et. al. 2015. Hey, you have given me too many knobs!: understanding and dealing with over-designed configuration in system software.FSE 2015
[2] Van Aken, Dana, et al. "Automatic Database Management System Tuning Through Large-scale Machine Learning." *International Conference on Management of Data*. ACM, 2017.

Performance Optimization is required since Default Configuration is Bad!

✓ Necessary

✓ Complex

Default **MySQL** configuration in 2016 assumes that machine has only 160 MB of RAM[1]

Rule-of-thumb settings for WordCount (in **Hadoop**) gave one of its worst execution times[2]

[1] Van Aken, Dana, et al. "Automatic Database Management System Tuning Through Large-scale Machine Learning." *International Conference on Management of Data*. ACM, 2017.
[2] Herodotou, Herodotos, et al. "Starfish: A Self-tuning System for Big Data Analytics." *CIDR*

Performance Optimization can be Expensive!

✓ Necessary

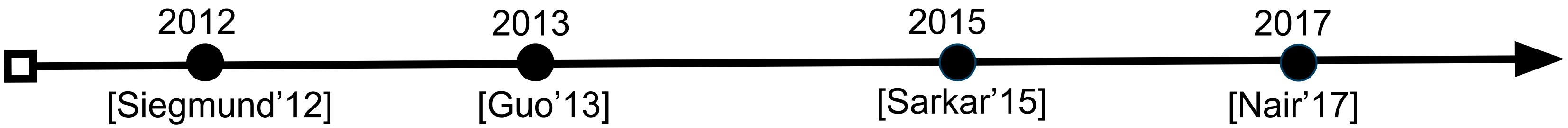
✓ Complex

✓ Default is bad

- Evaluation of single instance of their hardware/software co-design problem can take weeks[1]
- Rolling Sort use-case required 21 days, within a total experimental time of about 2.5 months[2]
- Test suite generation using Evolutionary Algorithm can take weeks[3]
- Image recognition workload and speech recognition workload, jobs ran for many hours or days[4]

[1] Zuluaga, Marcela, et al. "Active learning for multi-objective optimization." *International Conference on Machine Learning*. 2013.
[2] Jamshidi, Pooyan, and Giuliano Casale. "An uncertainty-aware approach to optimal configuration of stream processing systems."*MASCOTS-2016*
[3] Wang, Tiantian, et al. "Searching for better configurations: a rigorous approach to clone evaluation." *FSE-2013*
[4] Venkataraman, Shivaram, et al. "Ernest: Efficient Performance Prediction for Large-Scale Advanced Analytics." *NSDI*. 2016.

Existing Solutions



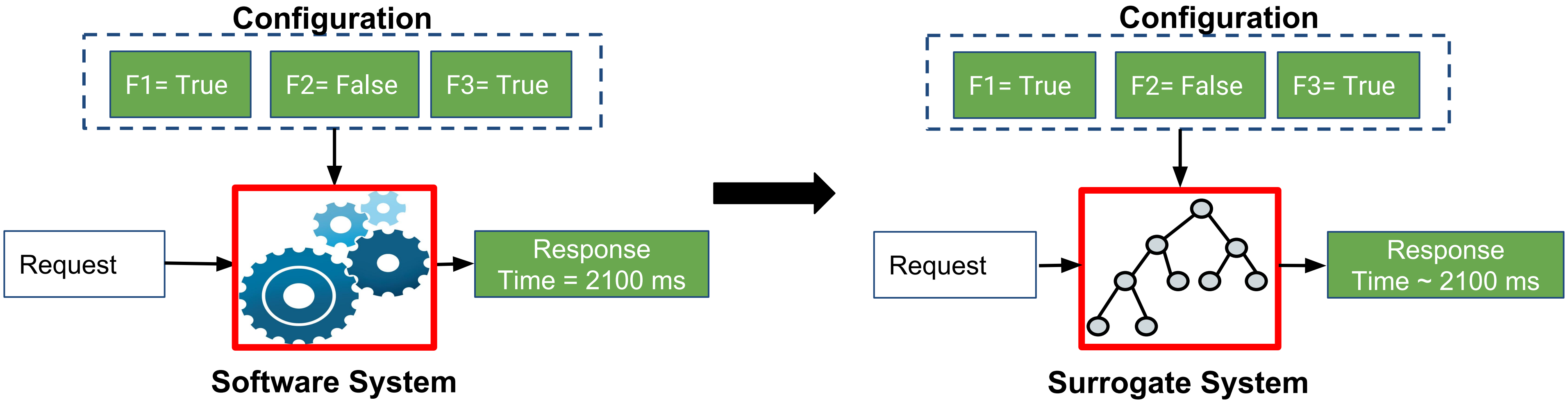
2012 [Siegmund'12]

2013 [Guo'13]

2015 [Sarkar'15]

2017 [Nair'17]

Accurately Model the configuration space



Configuration

F1= True

F2= False

F3= True

Request

Response Time = 2100 ms

Software System

Configuration

F1= True

F2= False

F3= True

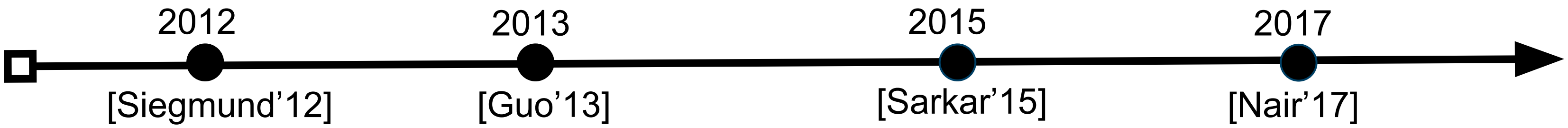
Request

Response Time ~ 2100 ms

Surrogate System

[Siegmund'12] Siegmund, Norbert, et al. "Predicting performance via automated feature-interaction detection." ICSE- 2012
[Guo'13] Guo, Jienmei, et al. "Variability-aware performance prediction: A statistical learning approach". ASE-2013
[Sarkar'15] Sarkar, Atri, et al. "Cost-efficient sampling for performance prediction of configurable systems (t)." ASE-2015
[Nair'17] Nair, Vivek, et al. "Faster discovery of faster system configurations with spectral learning." ASE Journal-2017 - to appear.

Existing Solutions


2012

[Siegmund'12]

2013

[Guo'13]

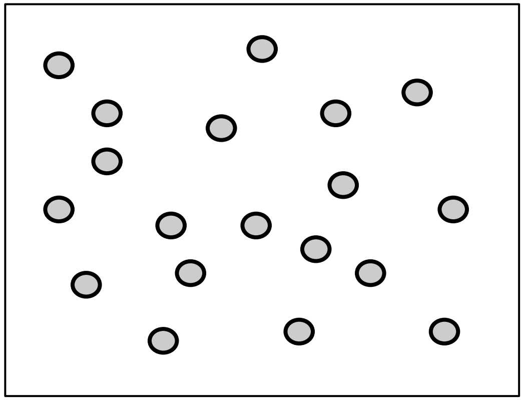
2015

[Sarkar'15]

2017

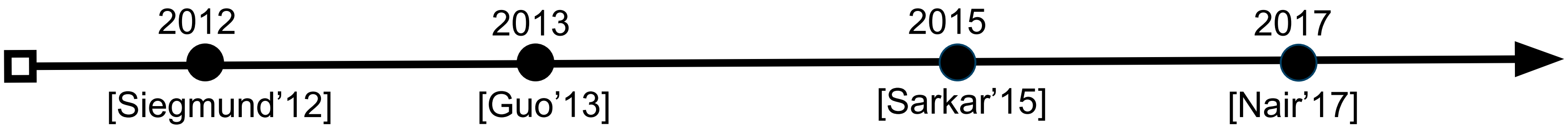
[Nair'17]

Accurately Model the configuration space

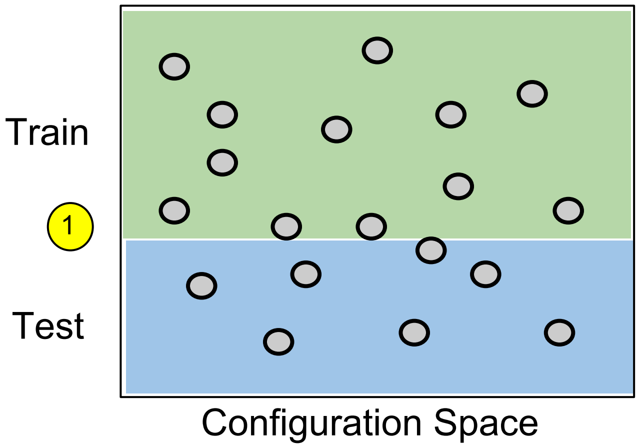

Configuration Space

[Siegmund'12] Siegmund, Norbert, et al. "Predicting performance via automated feature-interaction detection." ICSE- 2012
[Guo'13] Guo, Jienmei, et al. "Variability-aware performance prediction: A statistical learning approach". ASE-2013
[Sarkar'15] Sarkar, Atri, et al. "Cost-efficient sampling for performance prediction of configurable systems (t)." ASE-2015
[Nair'17] Nair, Vivek, et al. "Faster discovery of faster system configurations with spectral learning." ASE Journal-2017 - to appear.

Existing Solutions



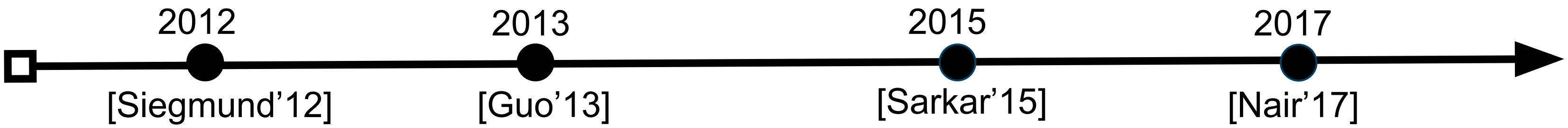
Accurately Model the configuration space



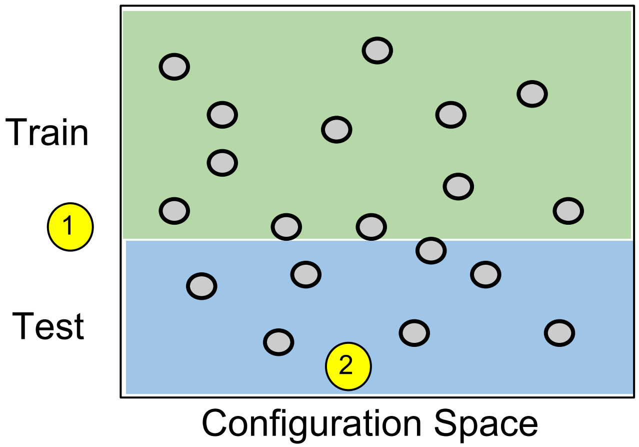
1. Divide the configuration space into *training* and *testing* sets;

[Siegmund'12] Siegmund, Norbert, et al. "Predicting performance via automated feature-interaction detection." ICSE- 2012
[Guo'13] Guo, Jienmei, et al. "Variability-aware performance prediction: A statistical learning approach". ASE-2013
[Sarkar'15] Sarkar, Atri, et al. "Cost-efficient sampling for performance prediction of configurable systems (t)." ASE-2015
[Nair'17] Nair, Vivek, et al. "Faster discovery of faster system configurations with spectral learning." ASE Journal-2017 - to appear.

Existing Solutions



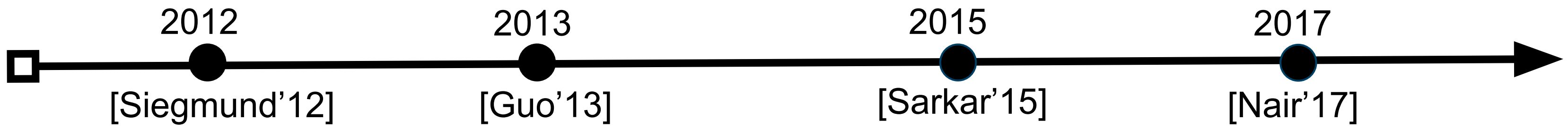
Accurately Model the configuration space



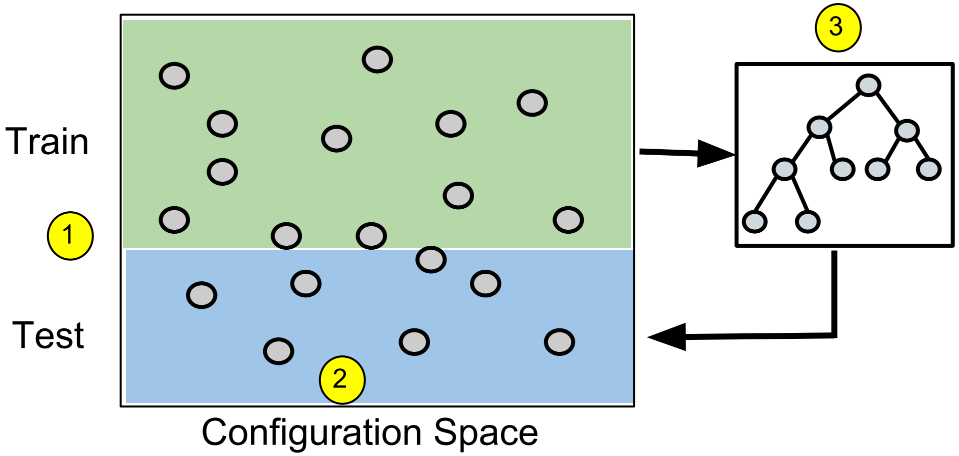
1. Divide the configuration space into *training* and *testing* sets;
2. Measure all the configurations in the *testing* set;

[Siegmund'12] Siegmund, Norbert, et al. "Predicting performance via automated feature-interaction detection." ICSE- 2012
[Guo'13] Guo, Jienmei, et al. "Variability-aware performance prediction: A statistical learning approach". ASE-2013
[Sarkar'15] Sarkar, Atri, et al. "Cost-efficient sampling for performance prediction of configurable systems (t)." ASE-2015
[Nair'17] Nair, Vivek, et al. "Faster discovery of faster system configurations with spectral learning." ASE Journal-2017 - to appear.

Existing Solutions



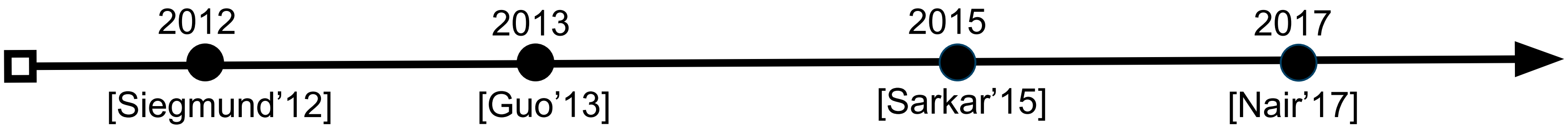
Accurately Model the configuration space



1. Divide the configuration space into *training* and *testing* sets;
2. Measure all the configurations in the *testing* set;
3. Iteratively sampling configuration from *training* set to build a *model* and test the model against *testing* set;

[Siegmund'12] Siegmund, Norbert, et al. "Predicting performance via automated feature-interaction detection." ICSE- 2012
[Guo'13] Guo, Jienmei, et al. "Variability-aware performance prediction: A statistical learning approach". ASE-2013
[Sarkar'15] Sarkar, Atri, et al. "Cost-efficient sampling for performance prediction of configurable systems (t)." ASE-2015
[Nair'17] Nair, Vivek, et al. "Faster discovery of faster system configurations with spectral learning." ASE Journal-2017 - to appear.

Existing Solutions


2012

[Siegmund'12]

2013

[Guo'13]

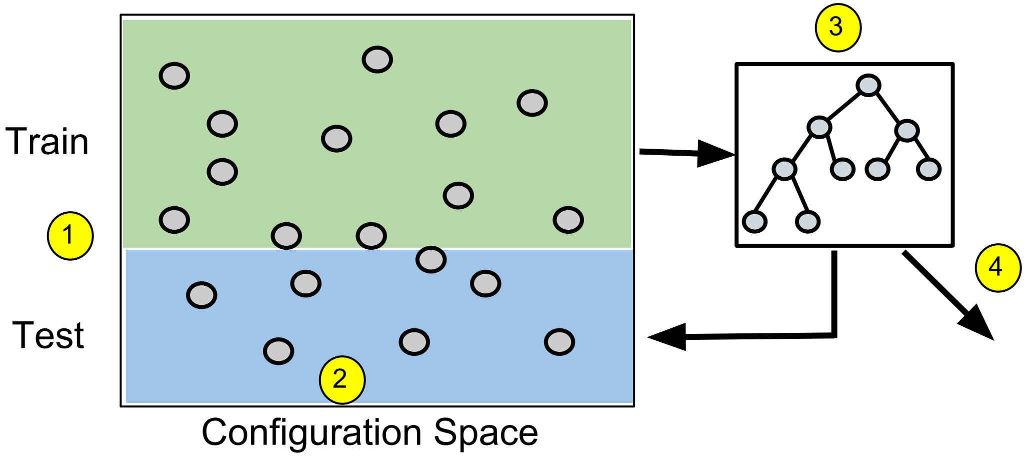
2015

[Sarkar'15]

2017

[Nair'17]

Accurately Model the configuration space



1. Divide the configuration space into *training* and *testing* sets;
2. Measure all the configurations in the *testing* set;
3. Iteratively sampling configuration from *training* set to build a *model* and test the model against *testing* set;
4. *Exit* when an accurate model is built (e.g., error = 0.1)

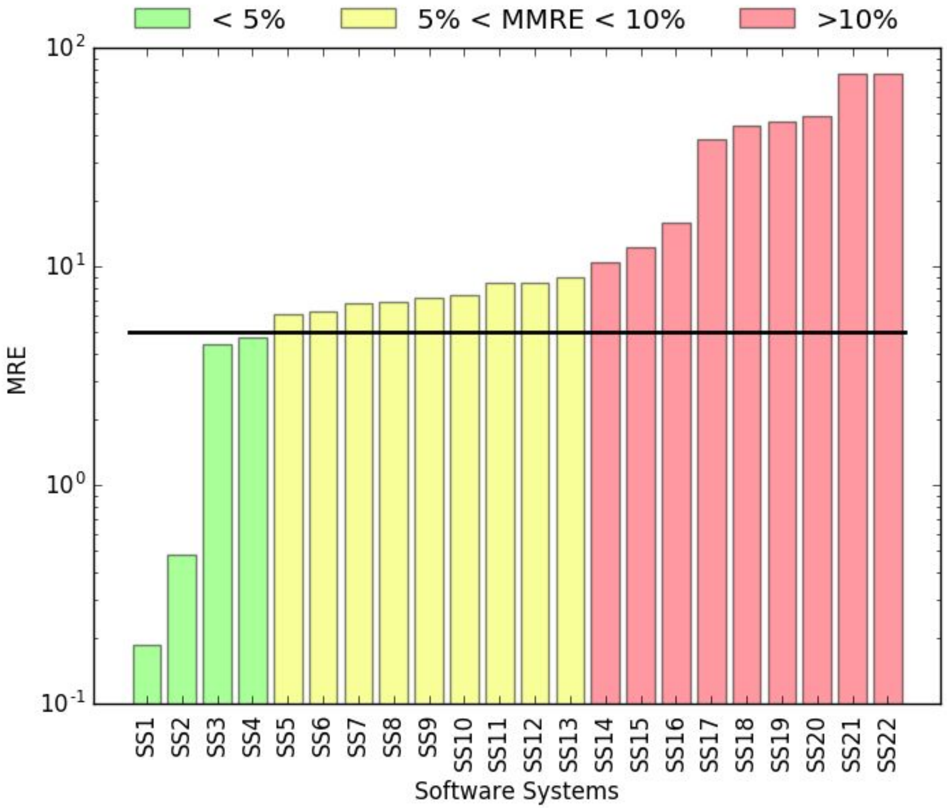
[Siegmund'12] Siegmund, Norbert, et al. "Predicting performance via automated feature-interaction detection." ICSE- 2012
[Guo'13] Guo, Jienmei, et al. "Variability-aware performance prediction: A statistical learning approach". ASE-2013
[Sarkar'15] Sarkar, Atri, et al. "Cost-efficient sampling for performance prediction of configurable systems (t)." ASE-2015
[Nair'17] Nair, Vivek, et al. "Faster discovery of faster system configurations with spectral learning." ASE Journal-2017 - to appear.

Limitation of Existing Solutions

Assumes, an Accurate Model of a software system can be built

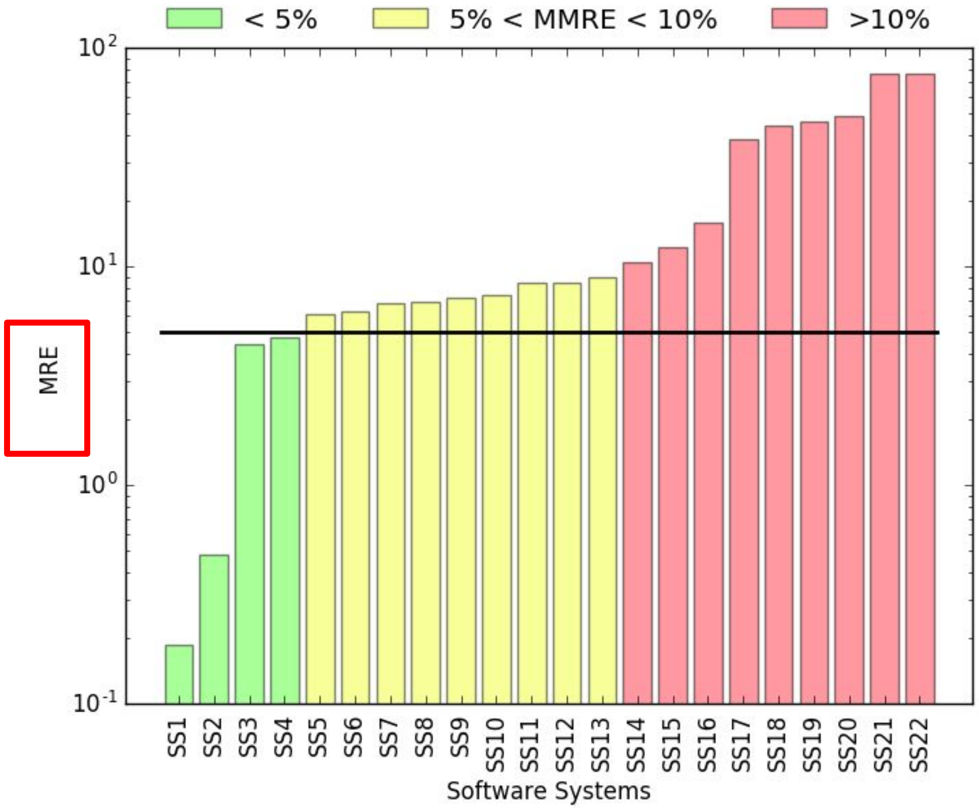
Limitation of Existing Solutions

Assumes, an Accurate Model of a software system can be built



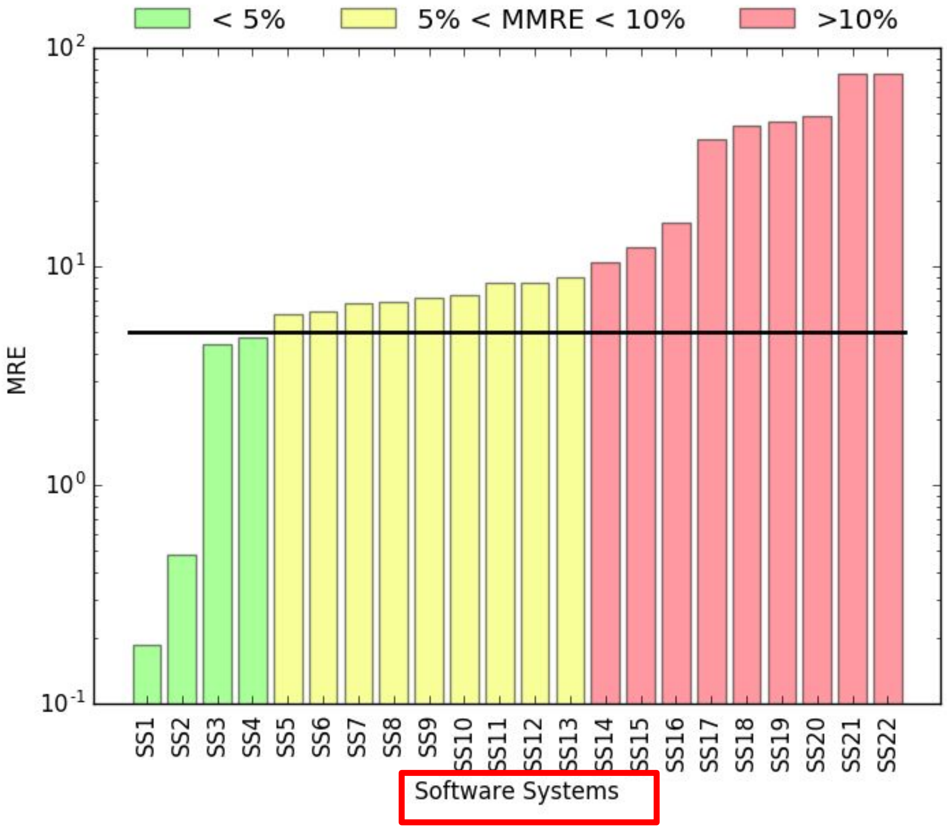
Limitation of Existing Solutions

Assumes, an Accurate Model of a software system can be built



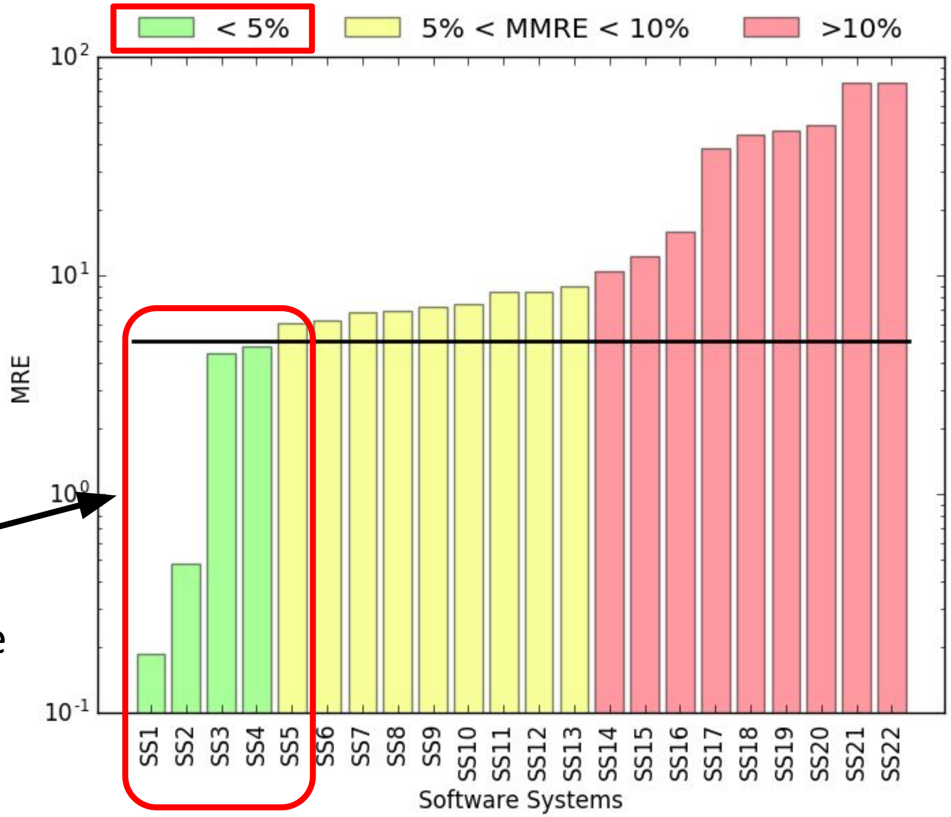
Limitation of Existing Solutions

Assumes, an Accurate Model of a software system can be built



Limitation of Existing Solutions

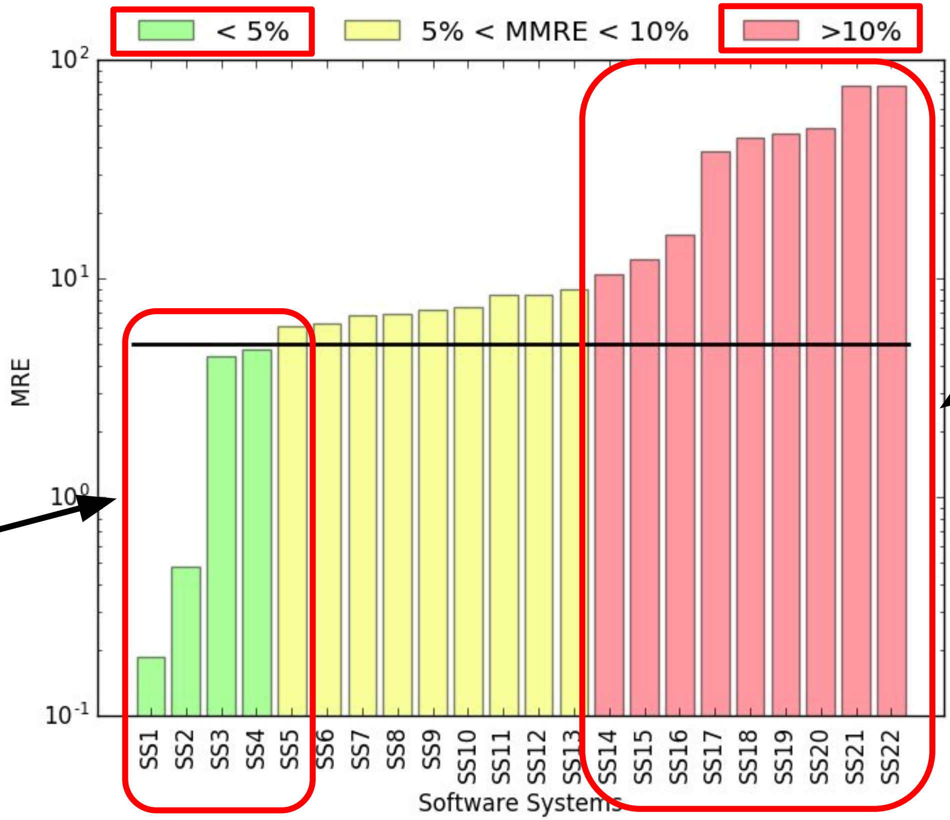
Assumes, an Accurate Model of a software system can be built



Software Systems used in existing works can be accurately modelled by CART

Limitation of Existing Solutions

Assumes, an Accurate Model of a software system can be built



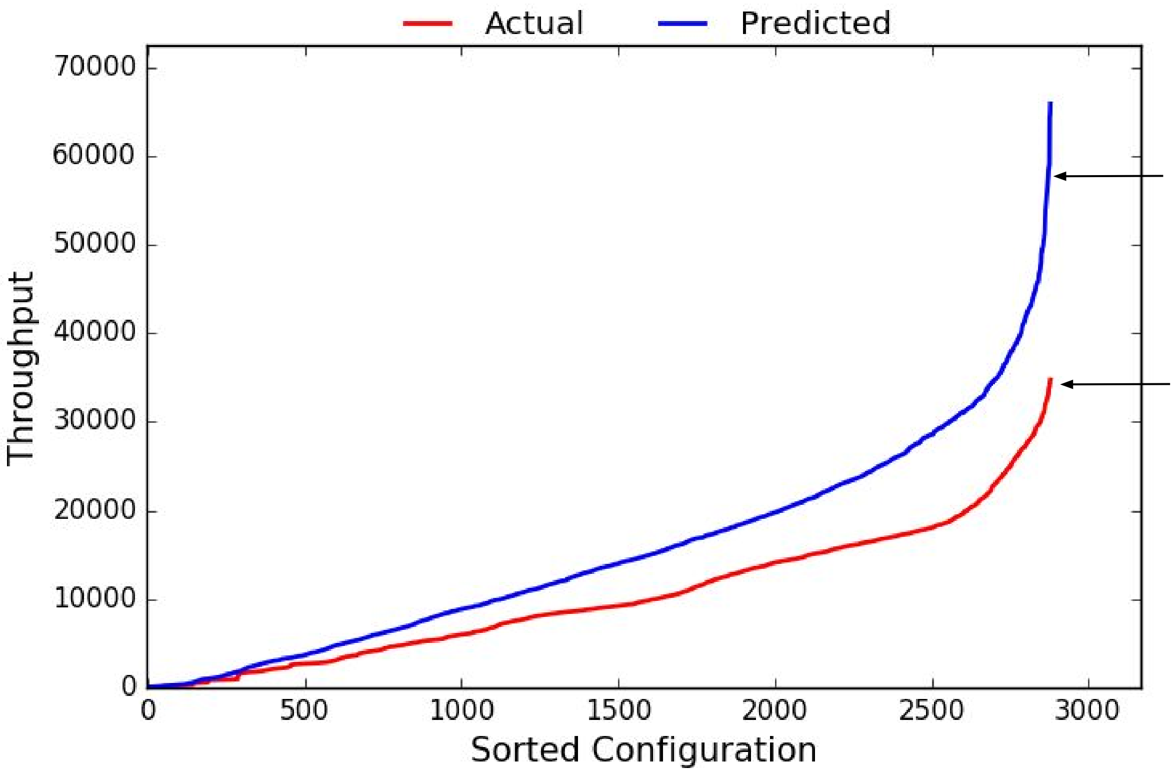
Some software systems cannot be accurately modelled

Software systems used in existing work can be accurately modelled

Core Insight

Rank-preserving model rather than highly accurate model

Rank Preserving Model

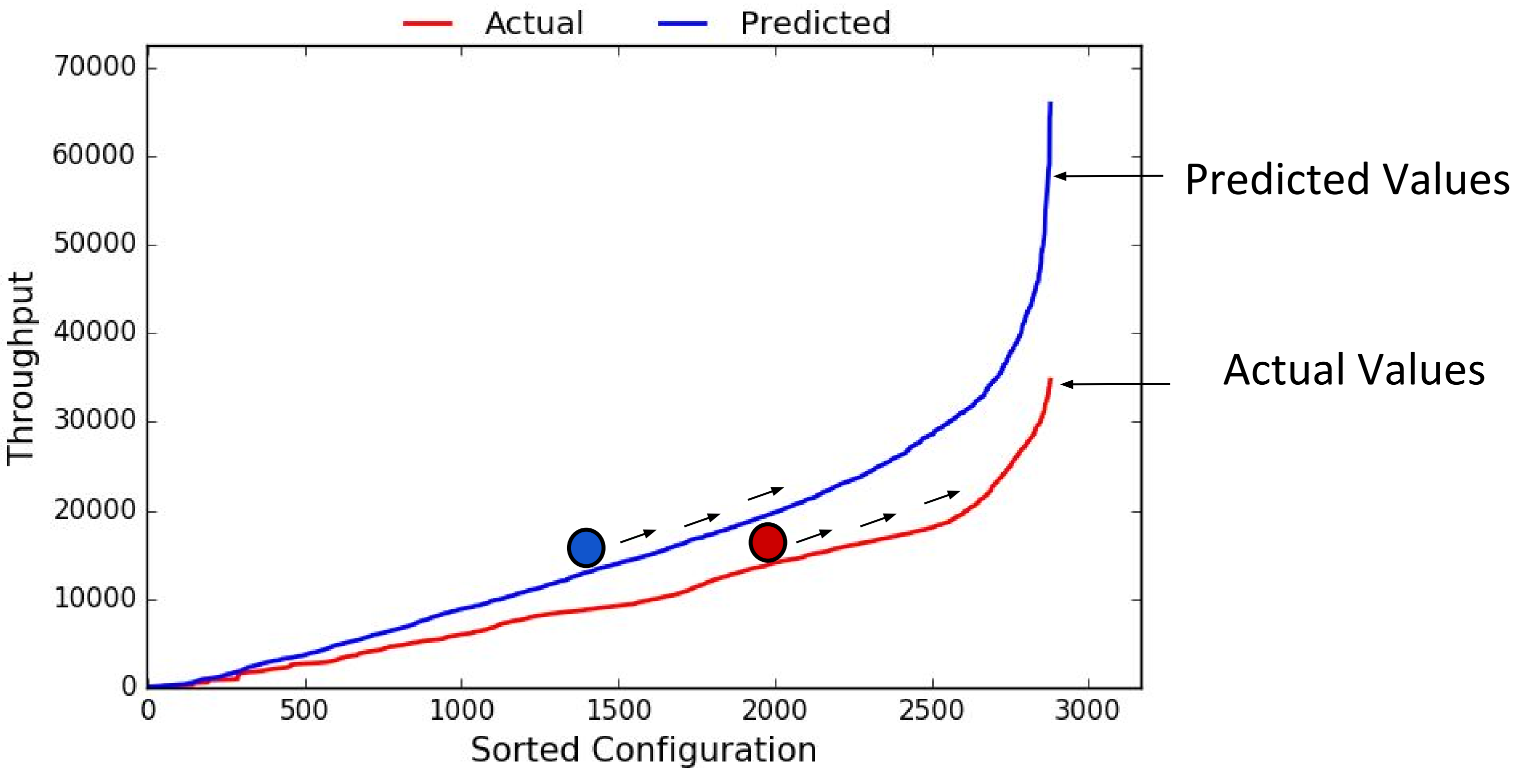


Predicted Values

Actual Values

Best Configuration obtained using actual and the predicted values is the same

Rank Preserving Model

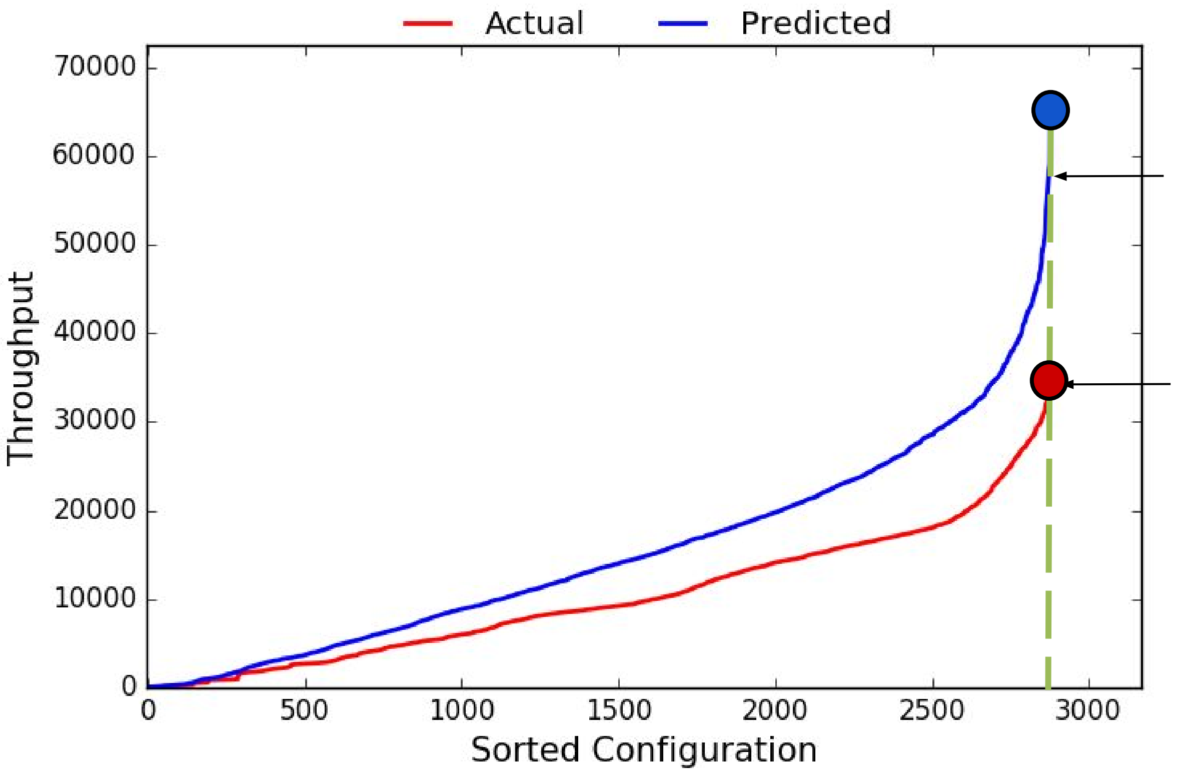


Predicted Values

Actual Values

Best Configuration obtained using actual and the predicted values is the same

Rank Preserving Model

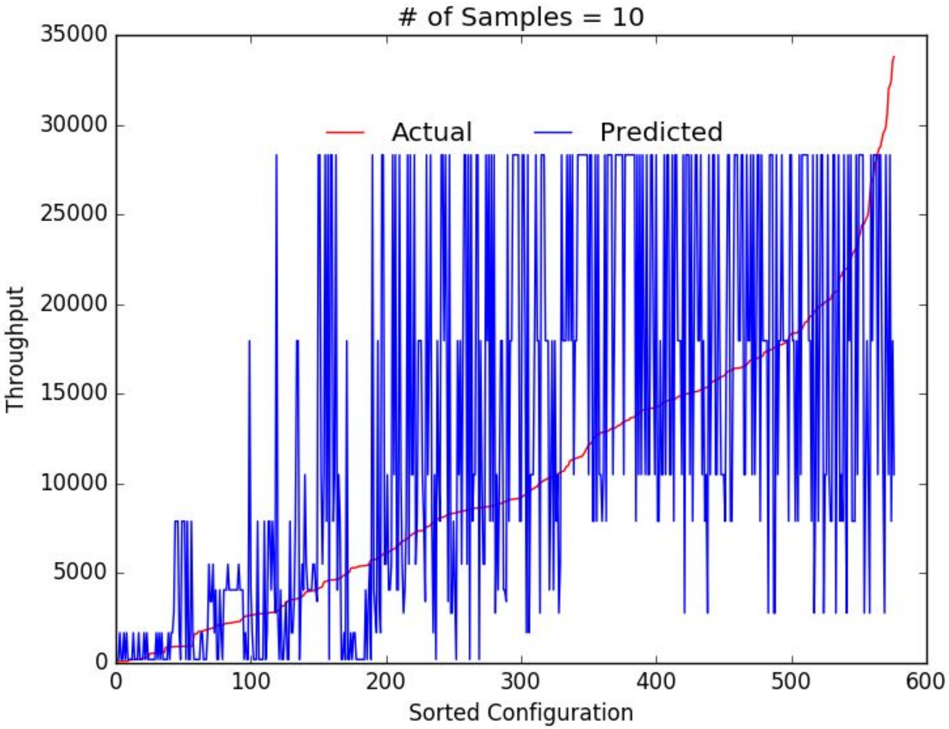


Predicted Values

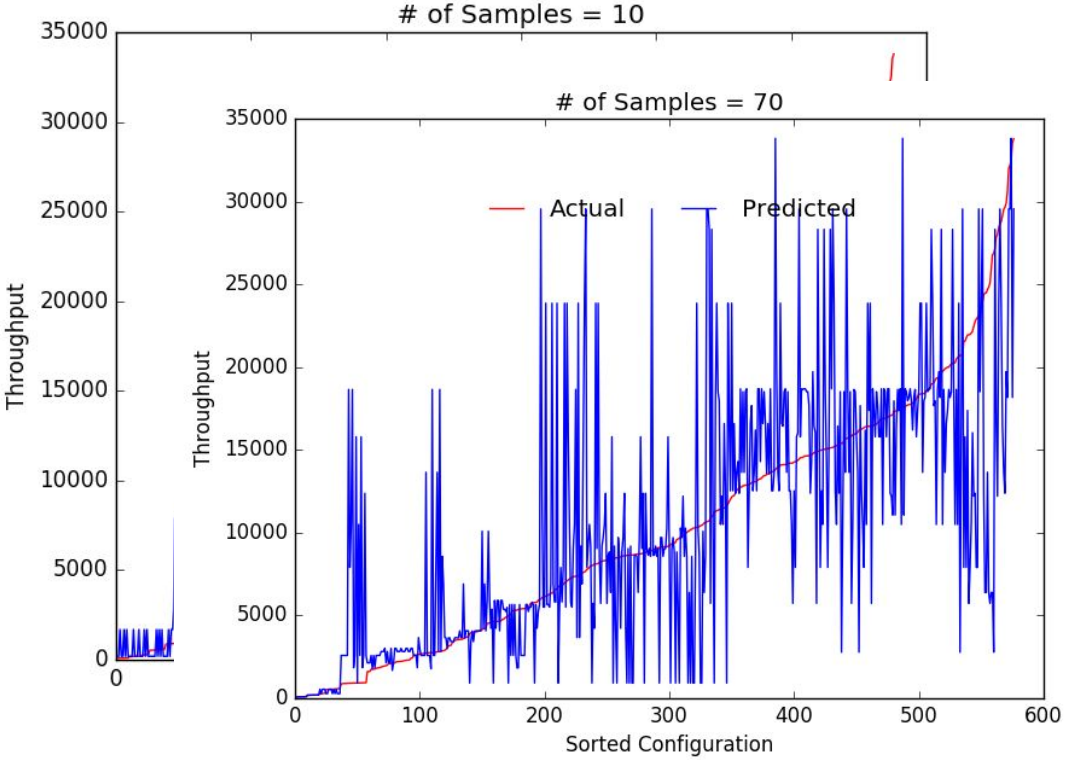
Actual Values

Best Configuration obtained using actual and the predicted values is the same

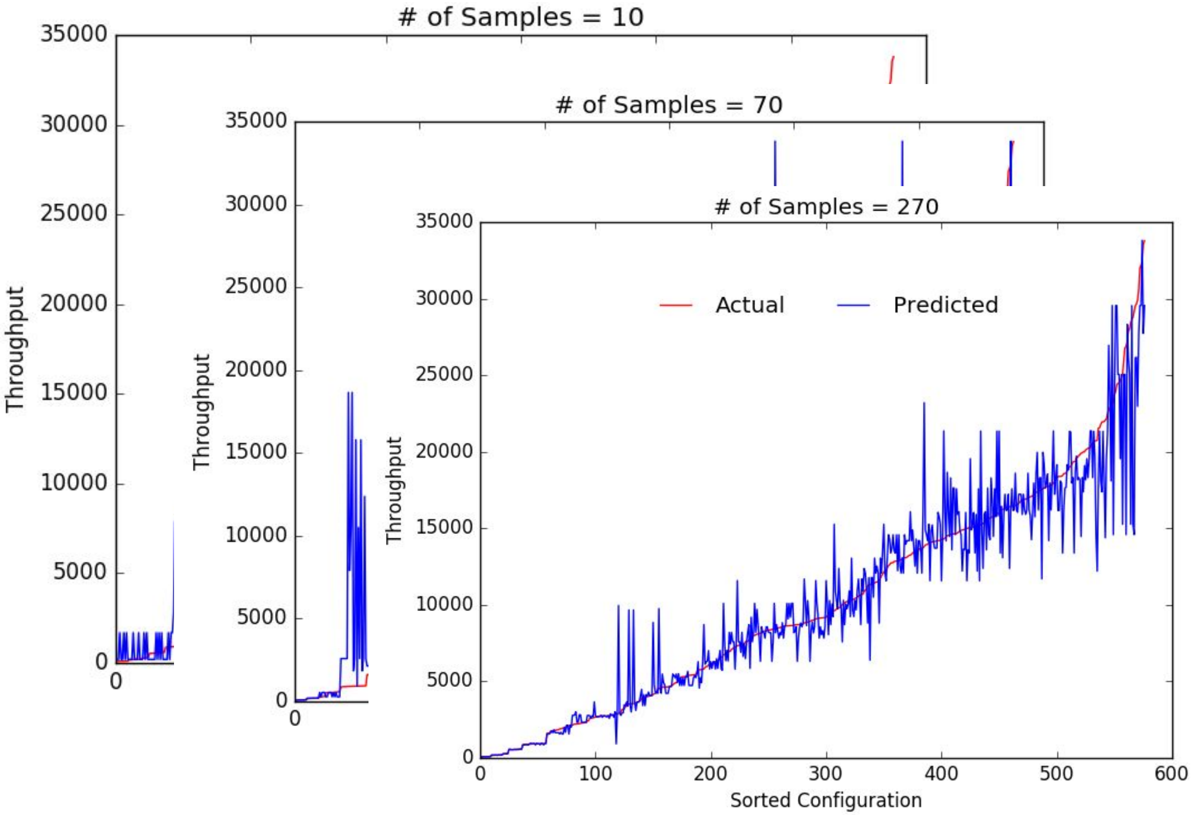
Rank Preserving Model



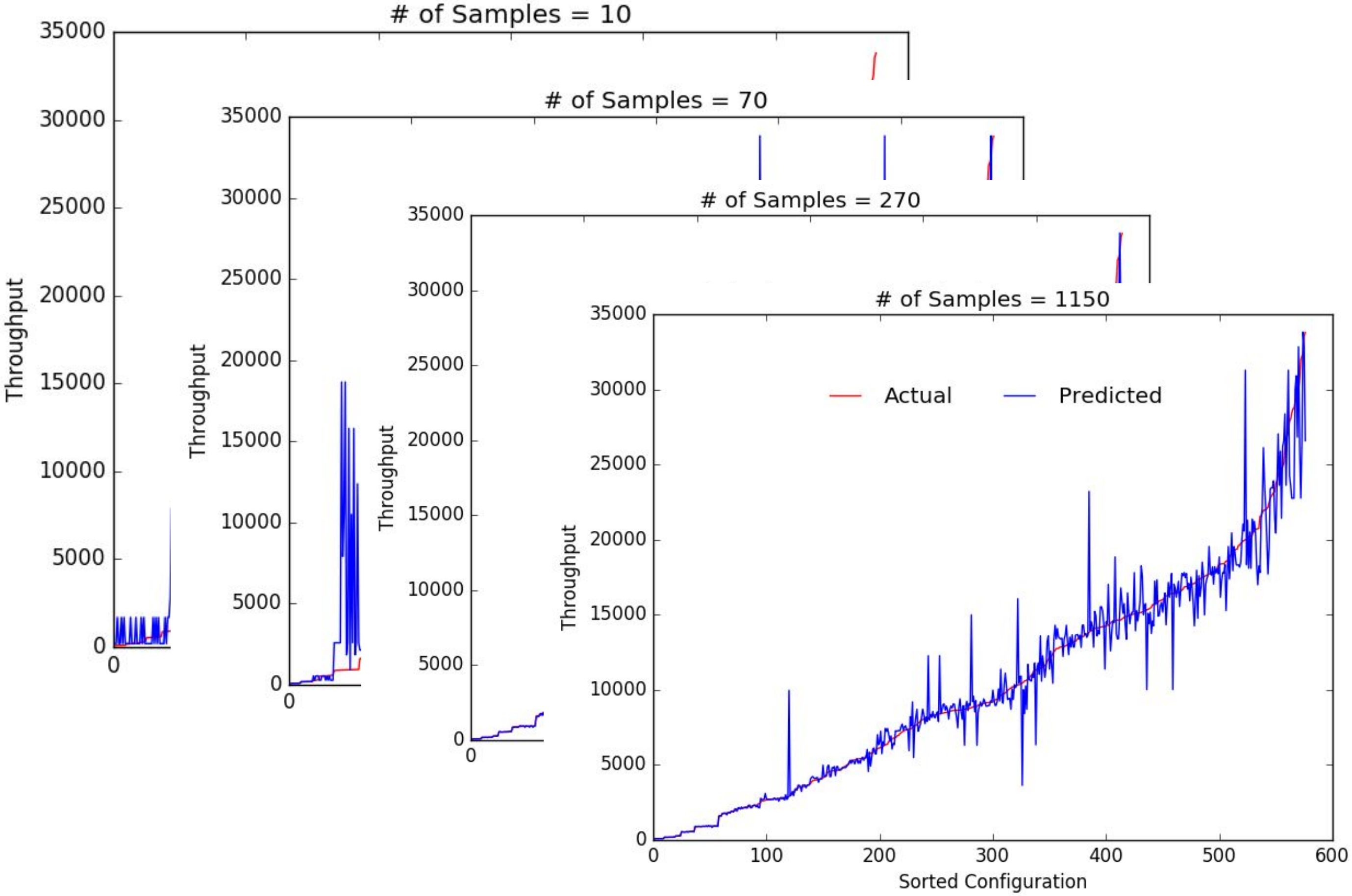
Rank Preserving Model



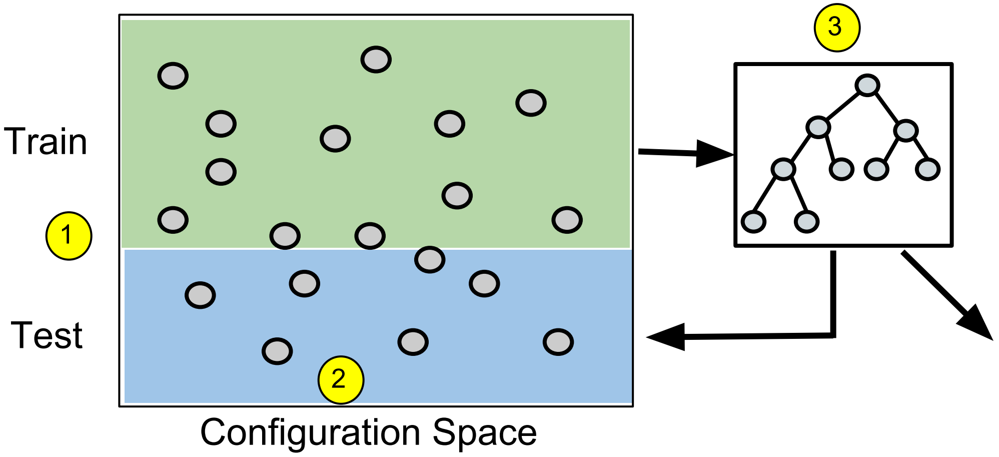
Rank Preserving Model



Rank Preserving Model

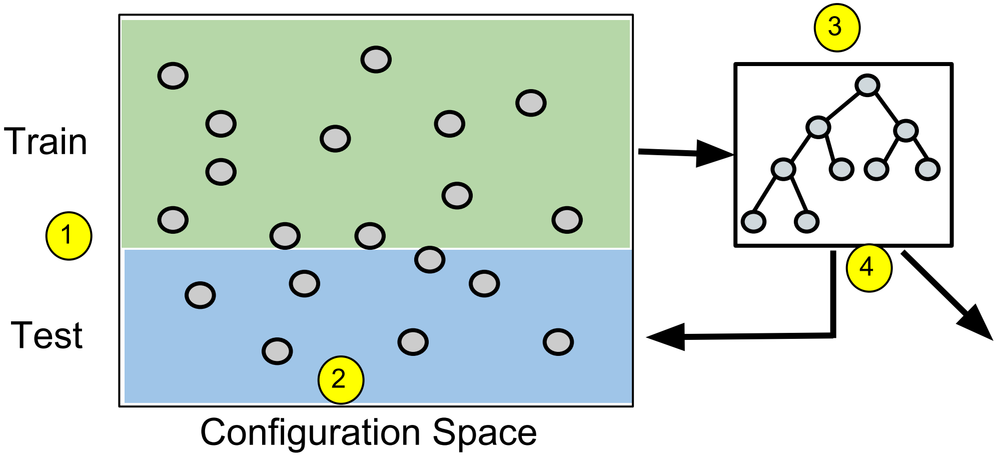


Rank Preserving Model



1. Divide the configuration space into *training* and *testing* sets;
2. Measure all the configurations in the *testing* set;
3. Iteratively sampling configuration from *training* set to build a *model* and test the model against *testing* set;

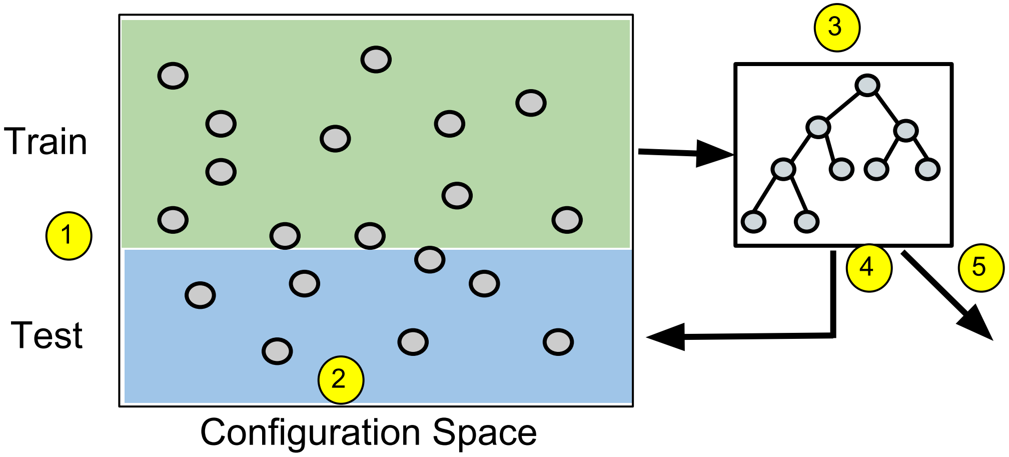
Rank Preserving Model



1. Divide the configuration space into *training* and *testing* sets;
2. Measure all the configurations in the *testing* set;
3. Iteratively sampling configuration from *training* set to build a *model* and test the model against *testing* set;
4. Calculate accuracy □ model should get progressively more accurate

$$accuracy = \frac{1}{n} \cdot \sum_{i=1}^{n} \left| rank(y_i) - rank(f(x_i)) \right|$$

Rank Preserving Model



1. Divide the configuration space into *training* and *testing* sets;
2. Measure all the configurations in the *testing* set;
3. Iteratively sampling configuration from *training* set to build a *model* and test the model against *testing* set;
4. Calculate accuracy model should get progressively more accurate
5. *Exit* when a model built does not improve (accuracy plateau)

$$accuracy = \frac{1}{n} \cdot \sum_{i=1}^{n} \left| rank(y_i) - rank(f(x_i)) \right|$$

Evaluation

Baselines

- Progressive Sampling[1]
  - Sequentially (randomly) sample configuration to build a decision tree till threshold accuracy is reached
- Projective Sampling[2]
  - Using minimal set of initial sample configurations to project the sampling cost based on a threshold accuracy

[1] Guo, Jianmei, et al. “Variability-aware performance prediction: A statistical learning approach”. ASE-2013
[2] Sarkar, Atri, et al. "Cost-efficient sampling for performance prediction of configurable systems (t)." ASE-2015

Research Questions

RQ1

- Can inaccurate models accurately rank configurations?

RQ2

- How expensive is a rank-based method?

Subject Software Systems

Video Encoder	X264		
Databases	ORACLE BERKELEY DB	?	Grid benchmark
Utility	GNU wget	Apache HTTP SERVER	Web server
Compression	?	APACHE STORM	Data processing
Numerical	Dune		

Subject Software Systems

Pooyan Jamshidi

Sven Apel

Norbert Siegmund

X264

ORACLE
BERKELEY DB

GNU wget

Dune

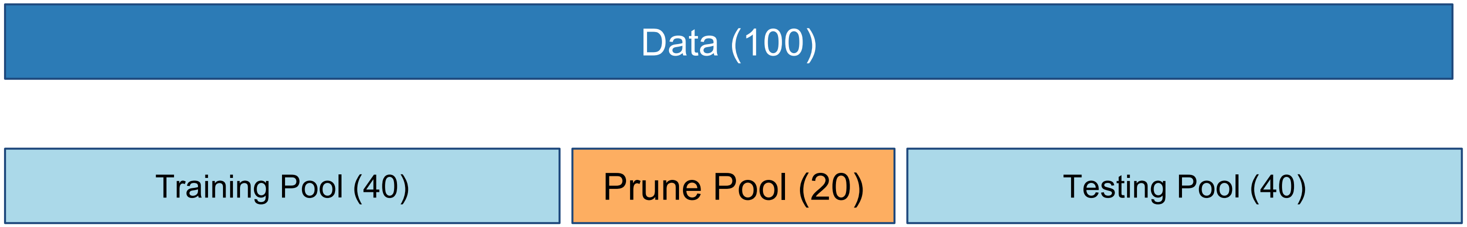
Apache
HTTP SERVER

APACHE
STORM

Combined effort = 6 computational months

Experimental Settings

Experimental Settings



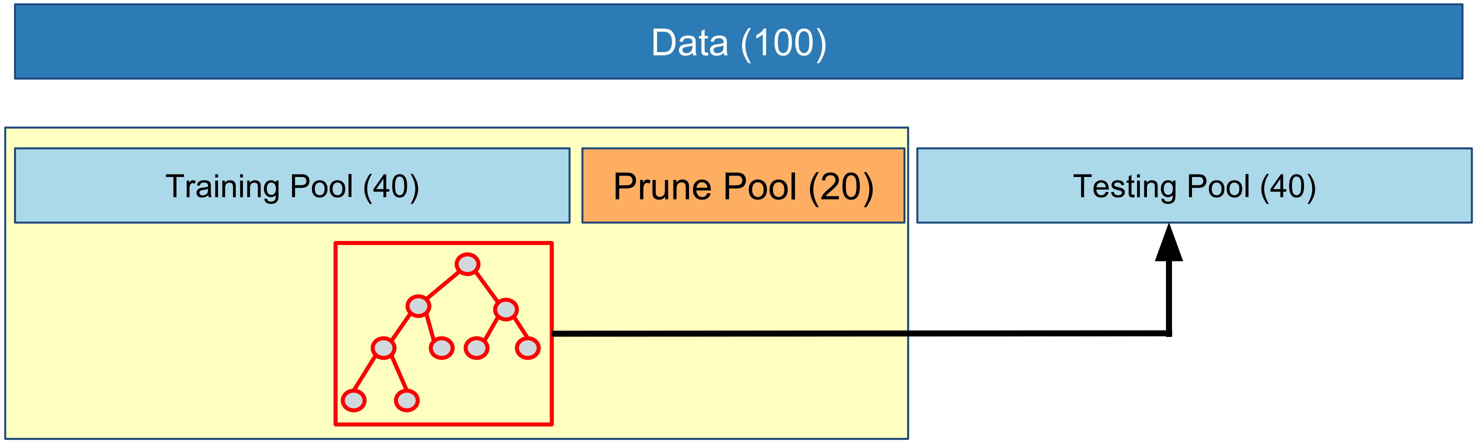
Data (100)

Training Pool (40)

Prune Pool (20)

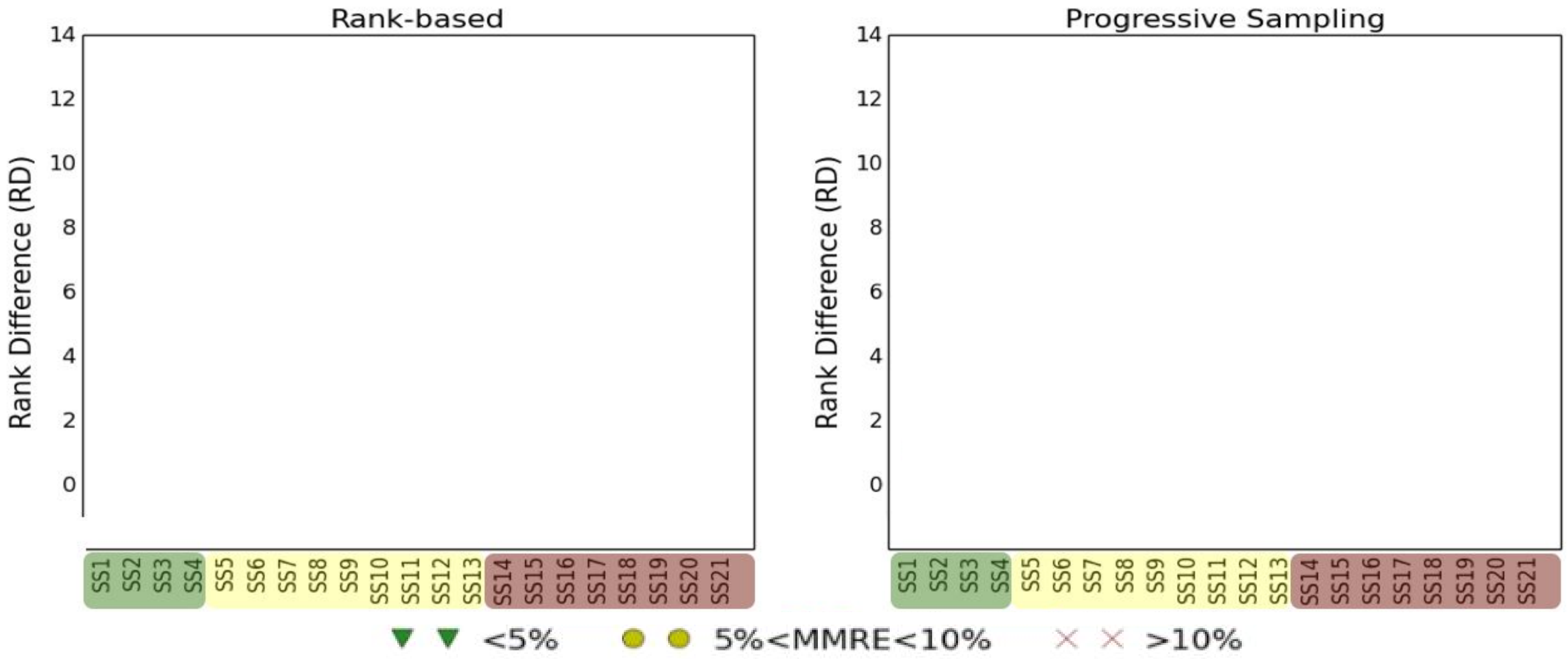
Testing Pool (40)

Experimental Settings

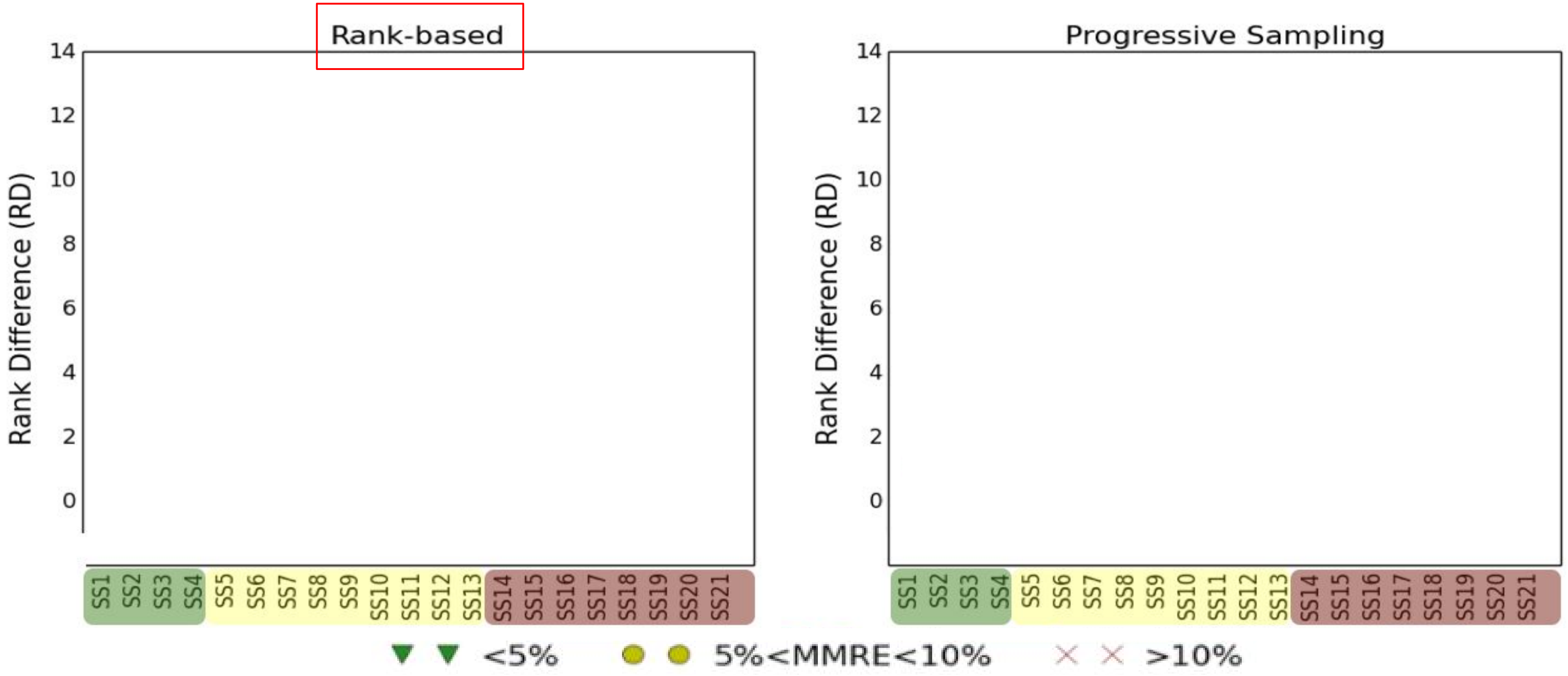


Results

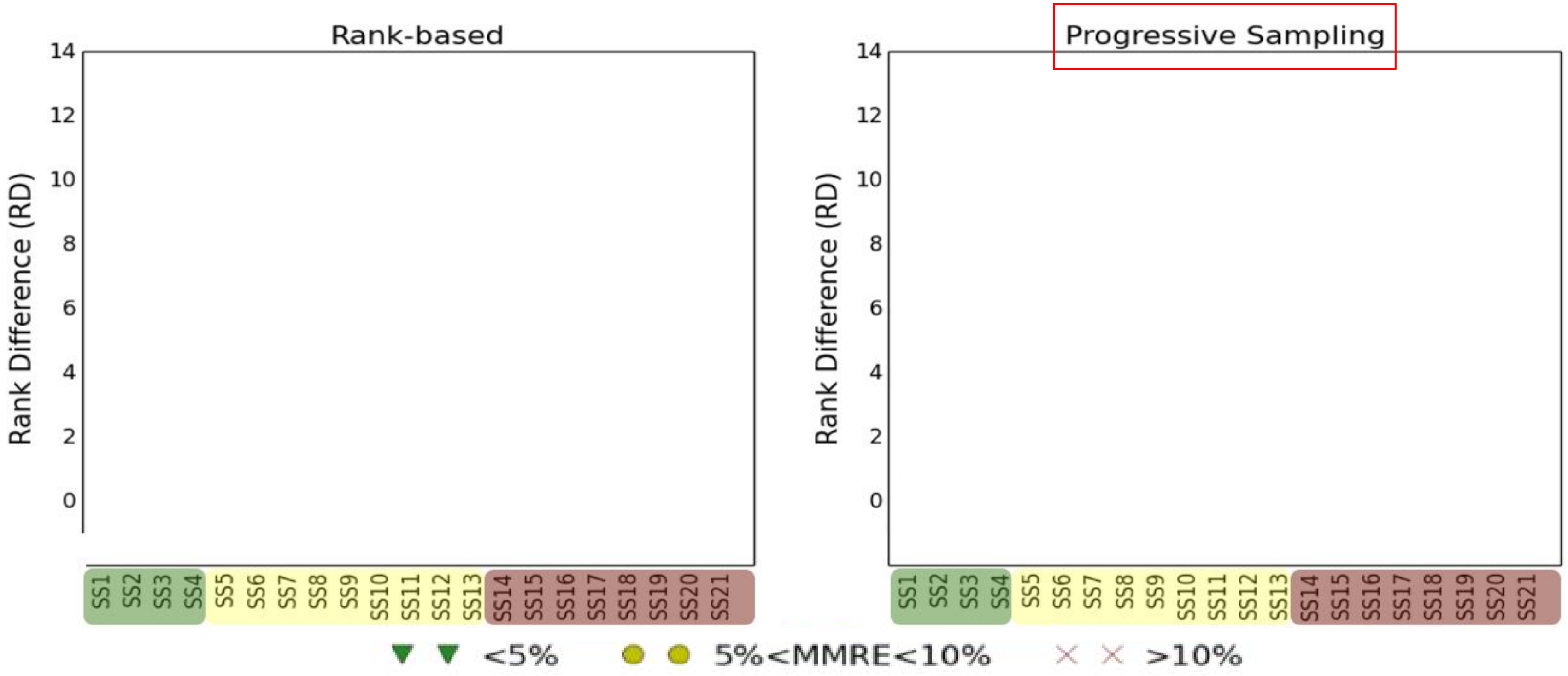
RQ1: Can inaccurate models accurately rank configurations?



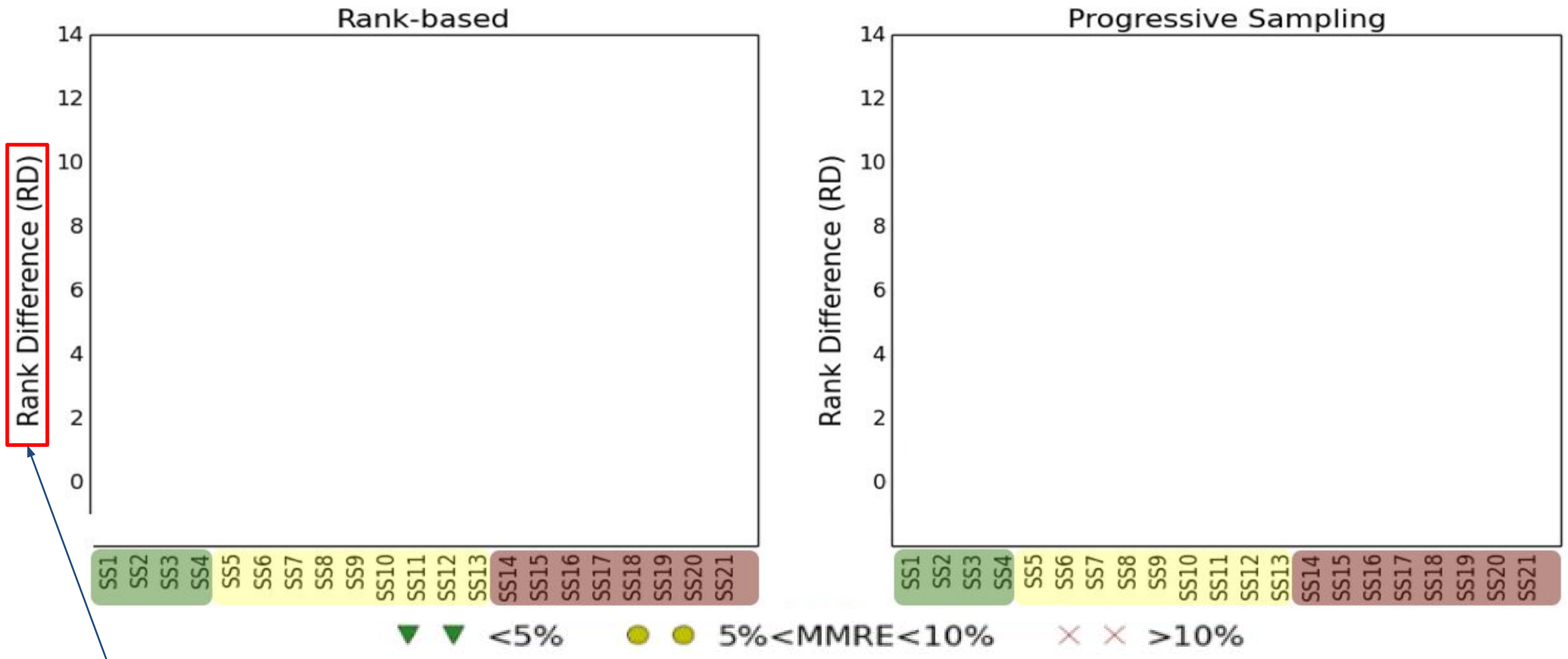
RQ1: Can inaccurate models accurately rank configurations?



RQ1: Can inaccurate models accurately rank configurations?



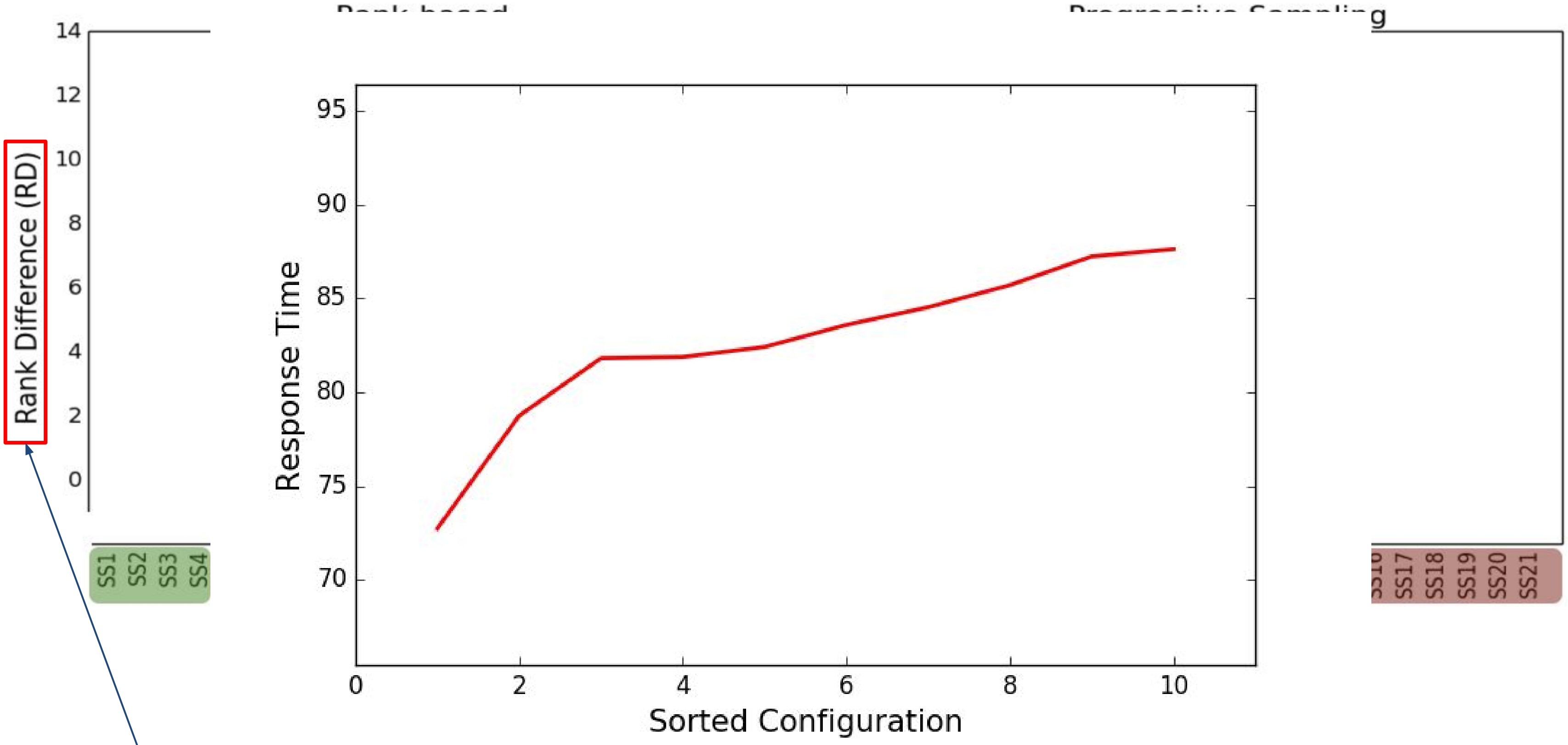
RQ1: Can inaccurate models accurately rank configurations?



The lower the better

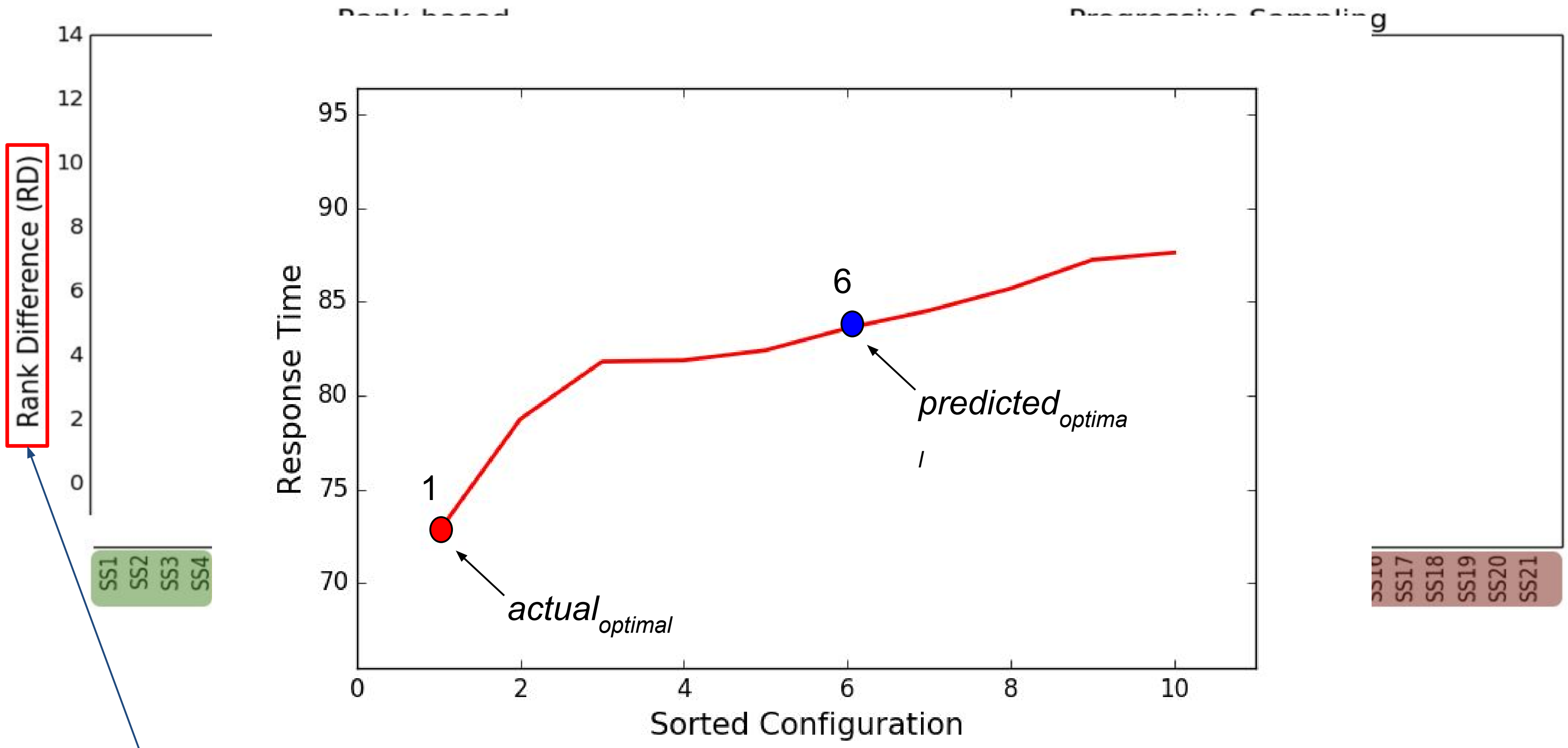
$$RD = |rank(actual_{optimal}) - rank(predicted_{optimal})|$$

RQ1: Can inaccurate models accurately rank configurations?



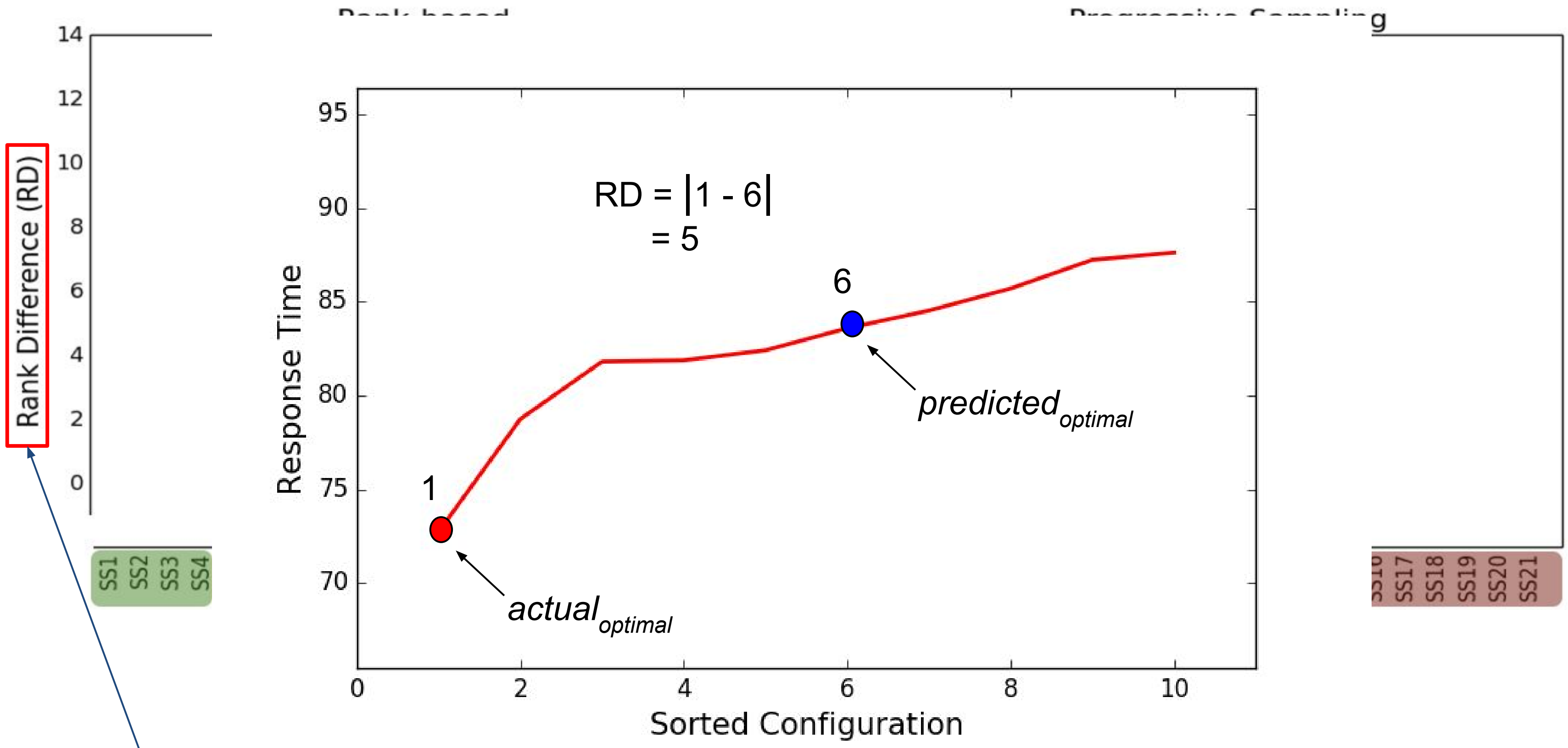
$$RD = |rank(actual_{optimal}) - rank(predicted_{optimal})|$$

RQ1: Can inaccurate models accurately rank configurations?



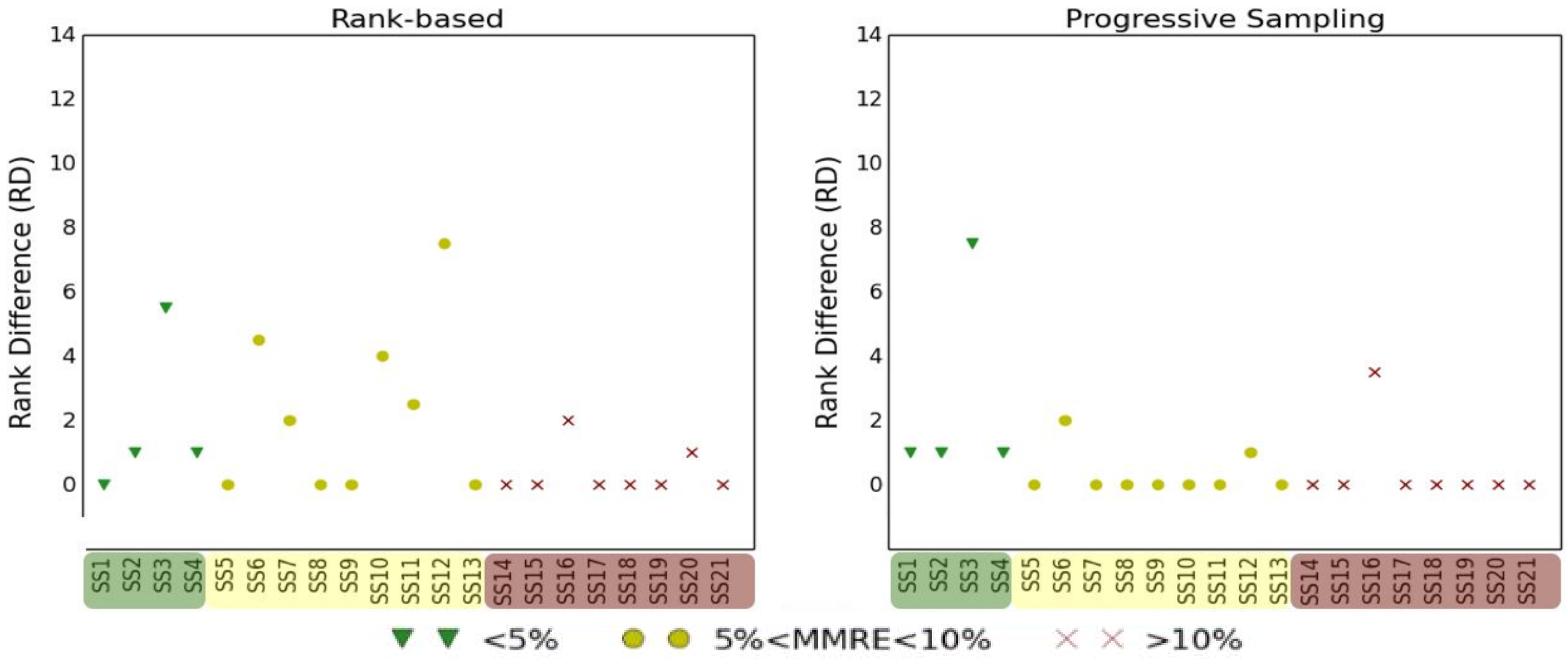
$$RD = |rank(actual_{optimal}) - rank(predicted_{optimal})|$$

RQ1: Can inaccurate models accurately rank configurations?

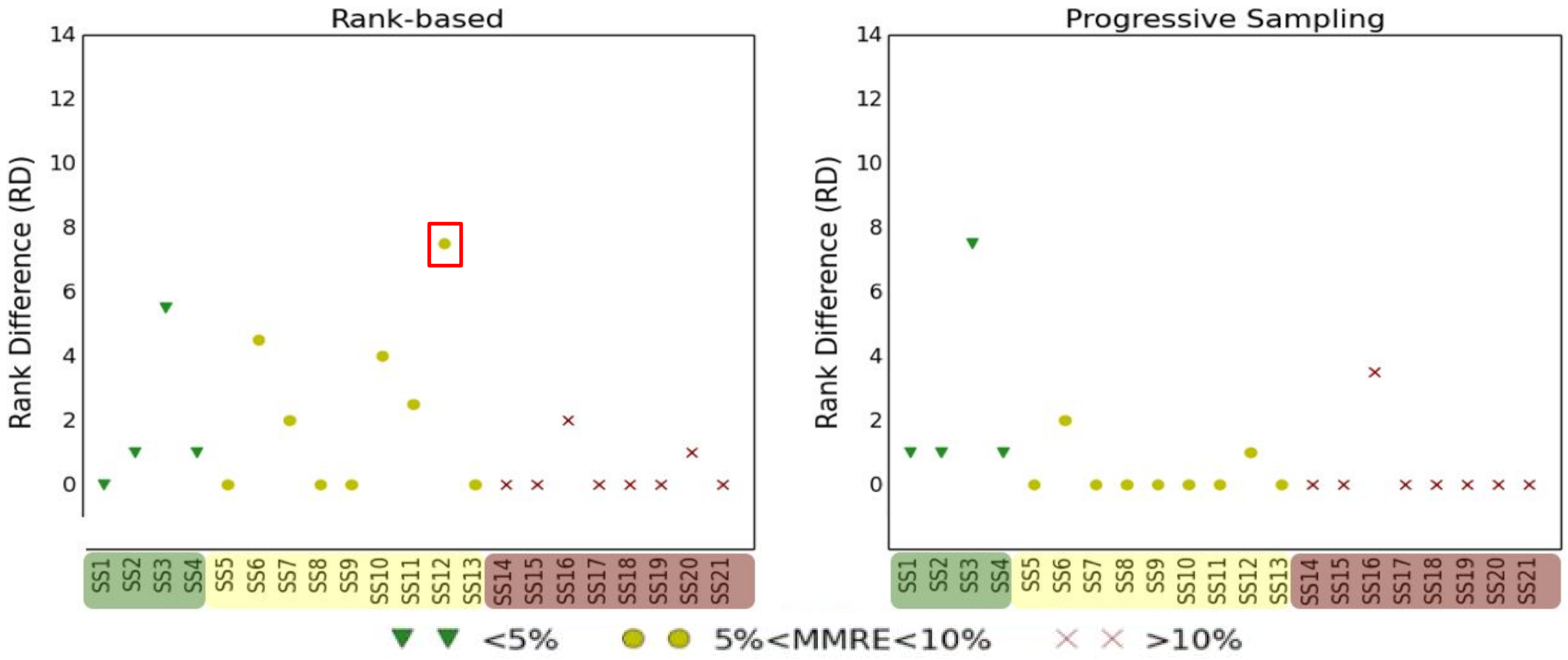


$$RD = |rank(actual_{optimal}) - rank(predicted_{optimal})|$$

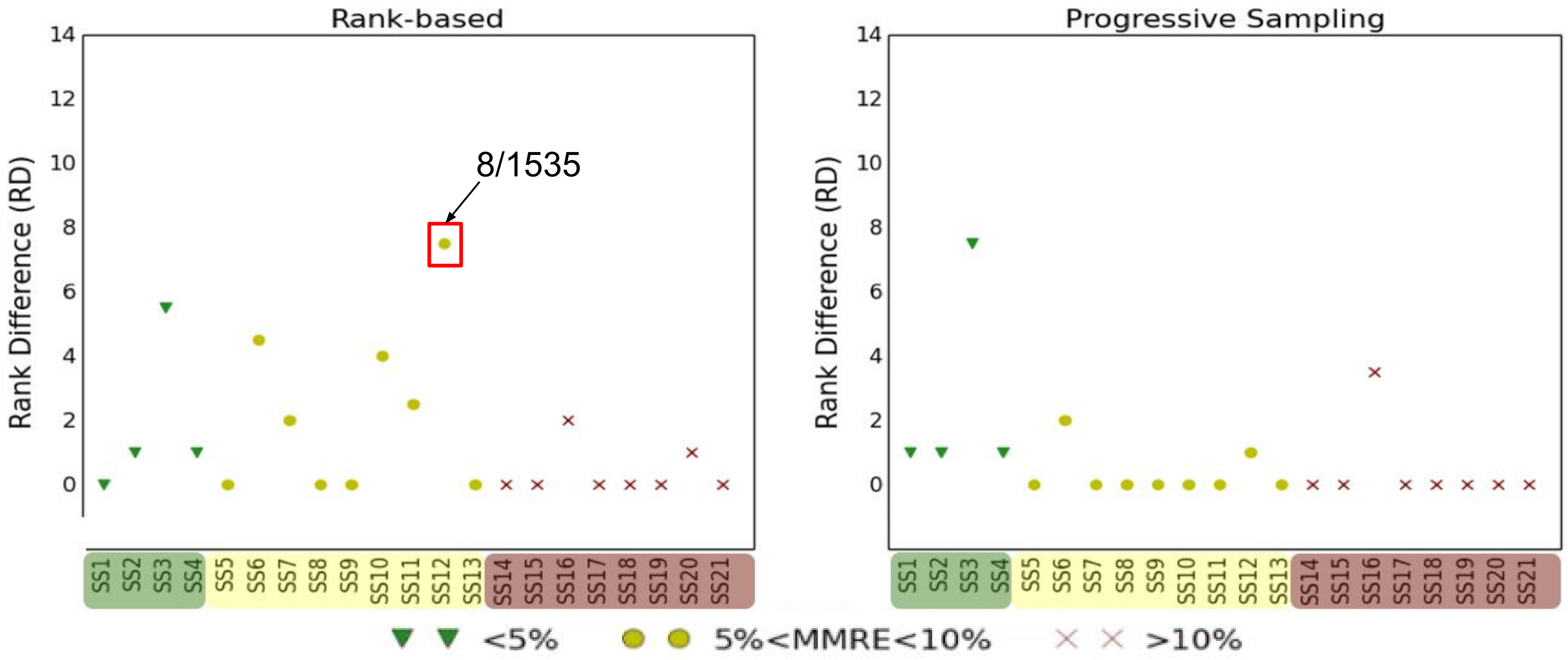
RQ1: Can inaccurate models accurately rank configurations?



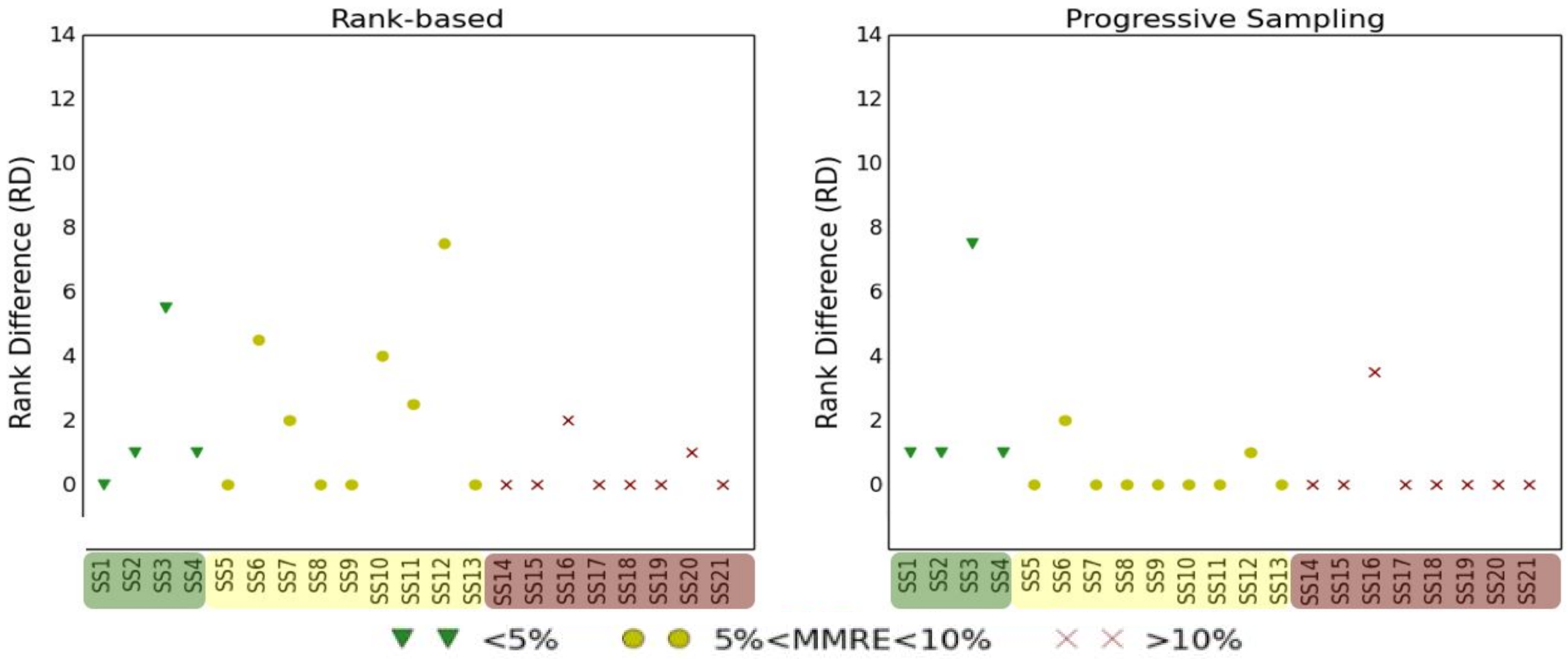
RQ1: Can inaccurate models accurately rank configurations?



RQ1: Can inaccurate models accurately rank configurations?

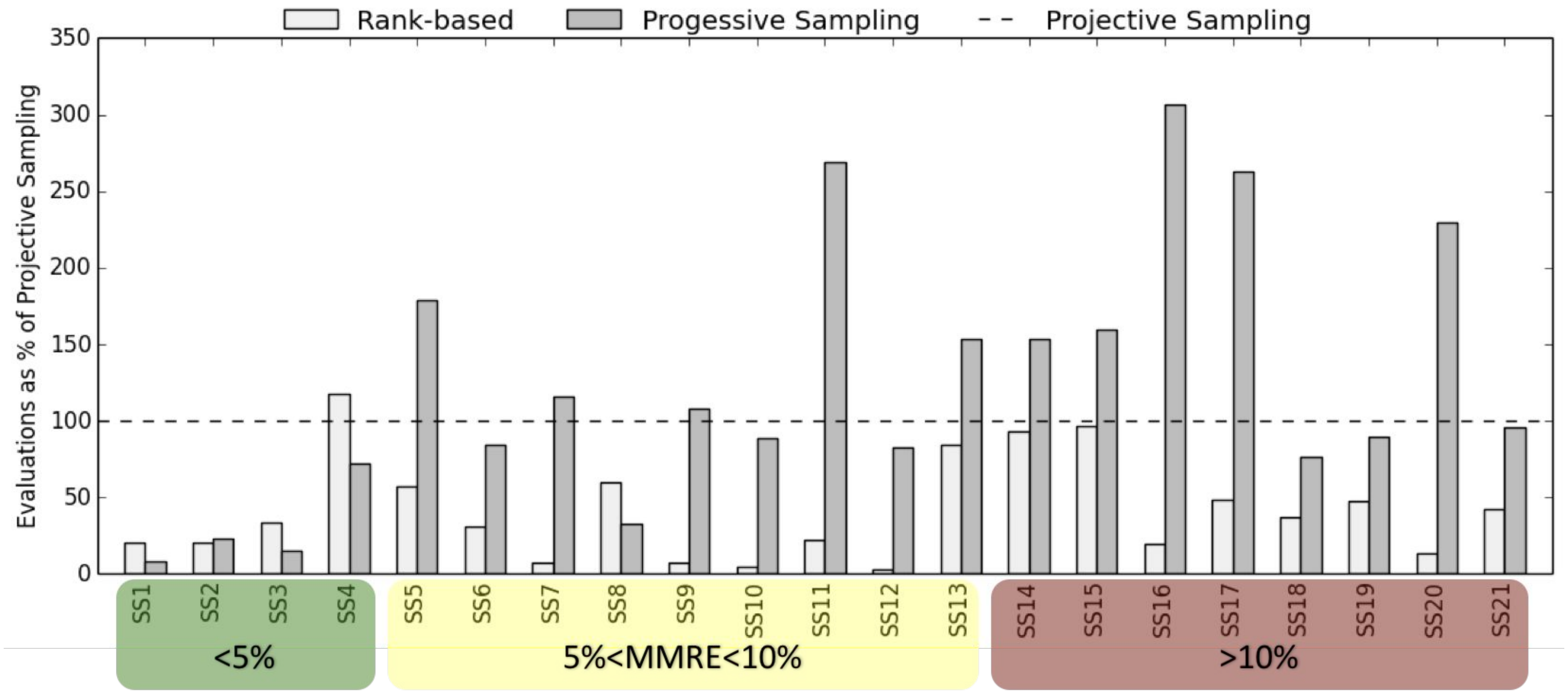


RQ1: Can inaccurate models accurately rank configurations?

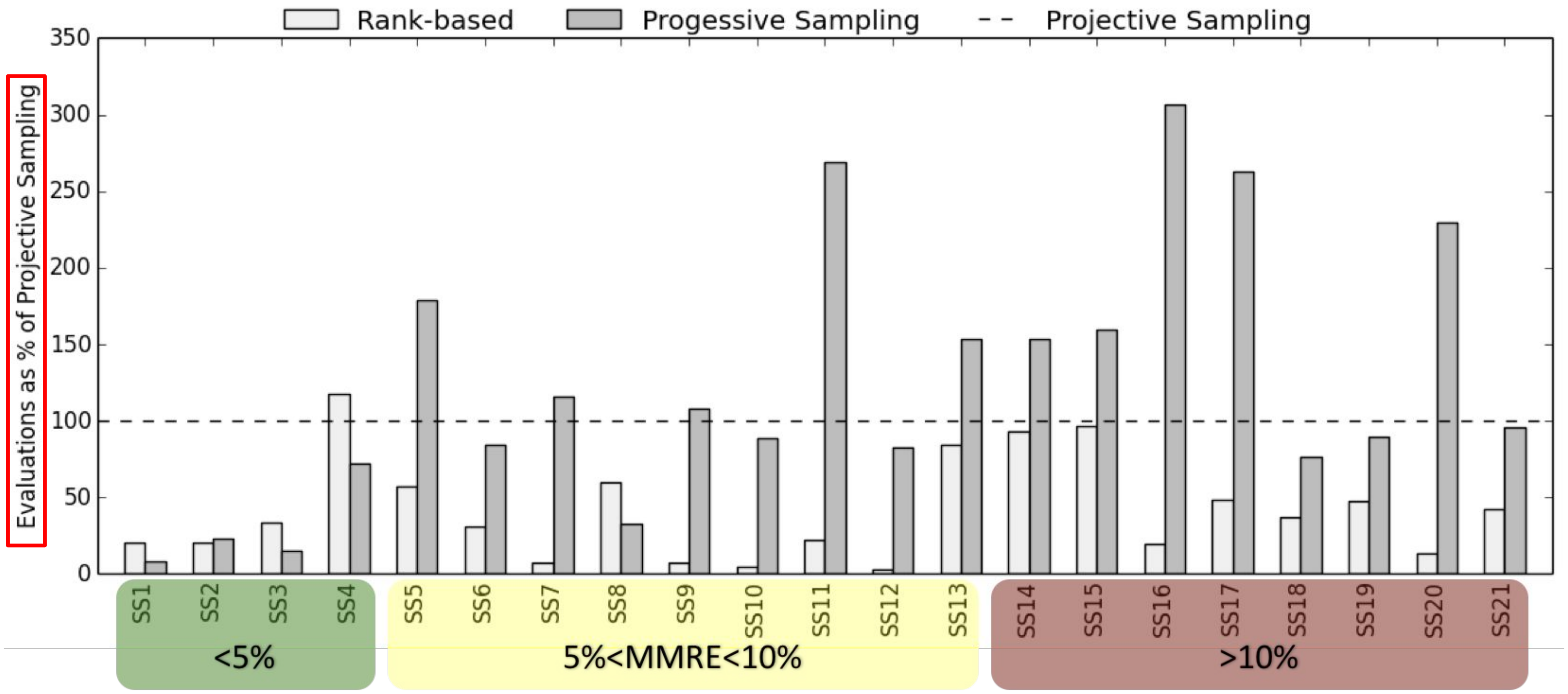


Yes, a rank preserving model can be useful in finding (near) optimal configurations!

RQ2: How expensive is a rank-based approach?

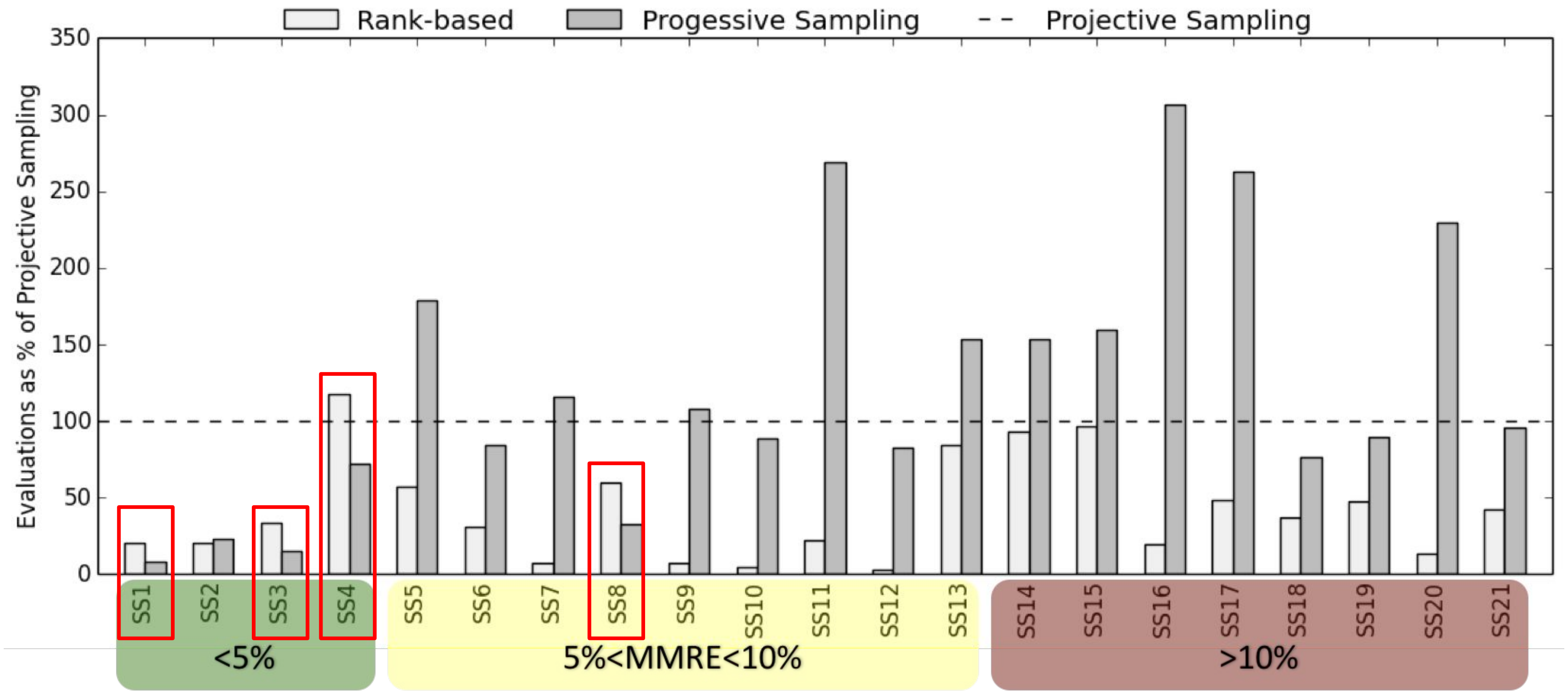


RQ2: How expensive is a rank-based approach?

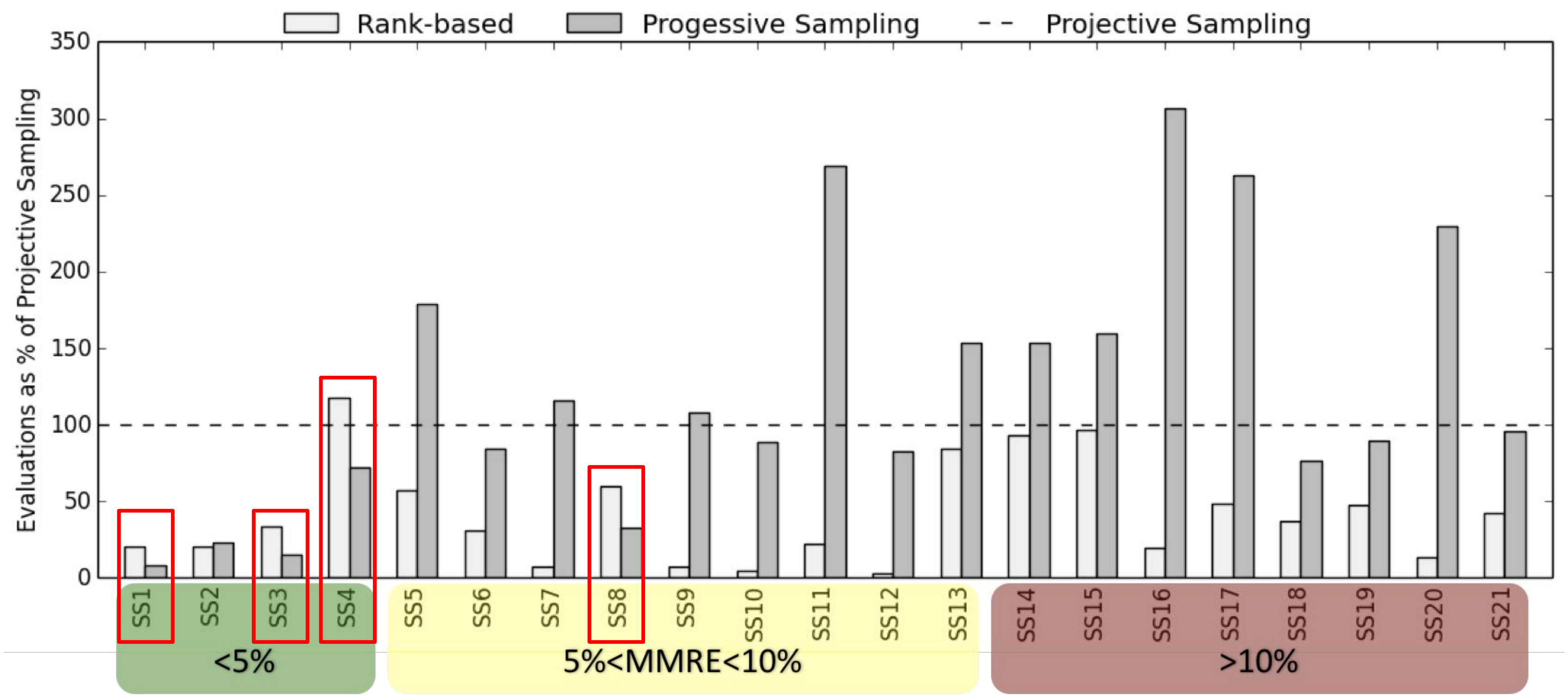


The lower the better

RQ2: How expensive is a rank-based approach?



RQ2: How expensive is a rank-based approach?



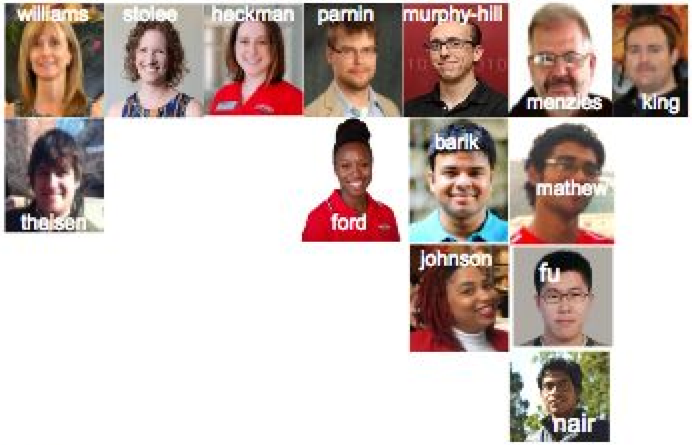
Yes, a rank-based approach requires fewer measurements!

Conclusion

- Rank-based method
  - a highly accurate model is not required for performance optimization;
  - performance optimization using predicted values correlated to actual values saves resources

- Future Work & Limitation
  - Relies heavily on testing pool (20%)
  - Bayesian based sequential sampling to reduce cost

NC STATE UNIVERSITY



vivekaxl@gmail.com

@vivekaxl

vivekaxl.com

Rank-preserving models

rather than

highly accurate models!

Expected Graduation: **May 2018**

Data Science, Performance Optimization, Evolutionary Algorithms

Bauhaus-
Universität
Weimar

UNIVERSITÄT
PASSAU