Challenges and Insights from Optimizing Configurable Software Systems

Norbert Siegmund

Bauhaus-Universität Weimar

2/6/2019

A Joint-Research Endeavor

Christian Kästner

Carnegie Mellon University

Pooyan Jamshidi

University of South Carolina

Miguel Velez

Carnegie Mellon University

Sven Apel

Alexander Grebhahn

Christian Kaltenecker

Florian Sattler

Universität Passau, Fakultät für Informatik und Mathematik

Jianmei Guo

Alibaba Group 阿里巴巴集团

Tim Menzies

NC State University

Vivek Nair

NC State University

Stefan Sobernig

WU Wirtschaftsuniversität Wien Vienna University of Economics and Business

Don Batory

The University of Texas at Austin

Jeho Oh

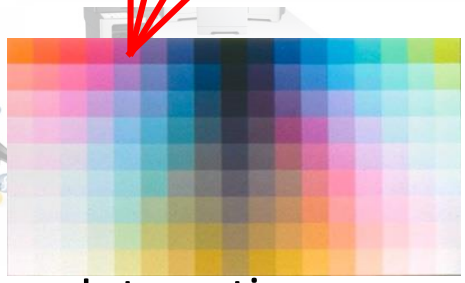
The University of Texas at Austin

Max Weber, Johannes Dorn, Nicolai Ruckel, Stefan Mühlbauer, André Karge

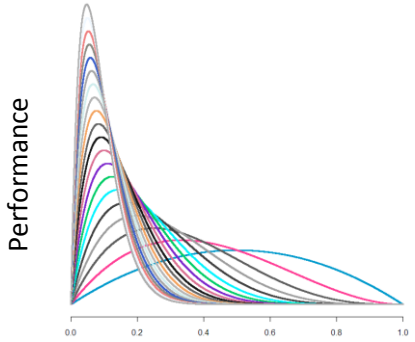
Bauhaus-Universität Weimar

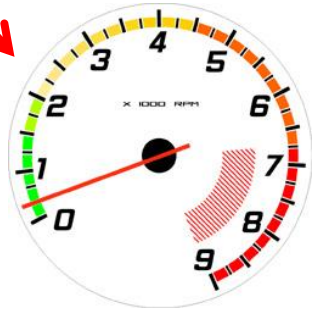
+ others who I forgot… sorry!

Configurable Software Systems



Interactions

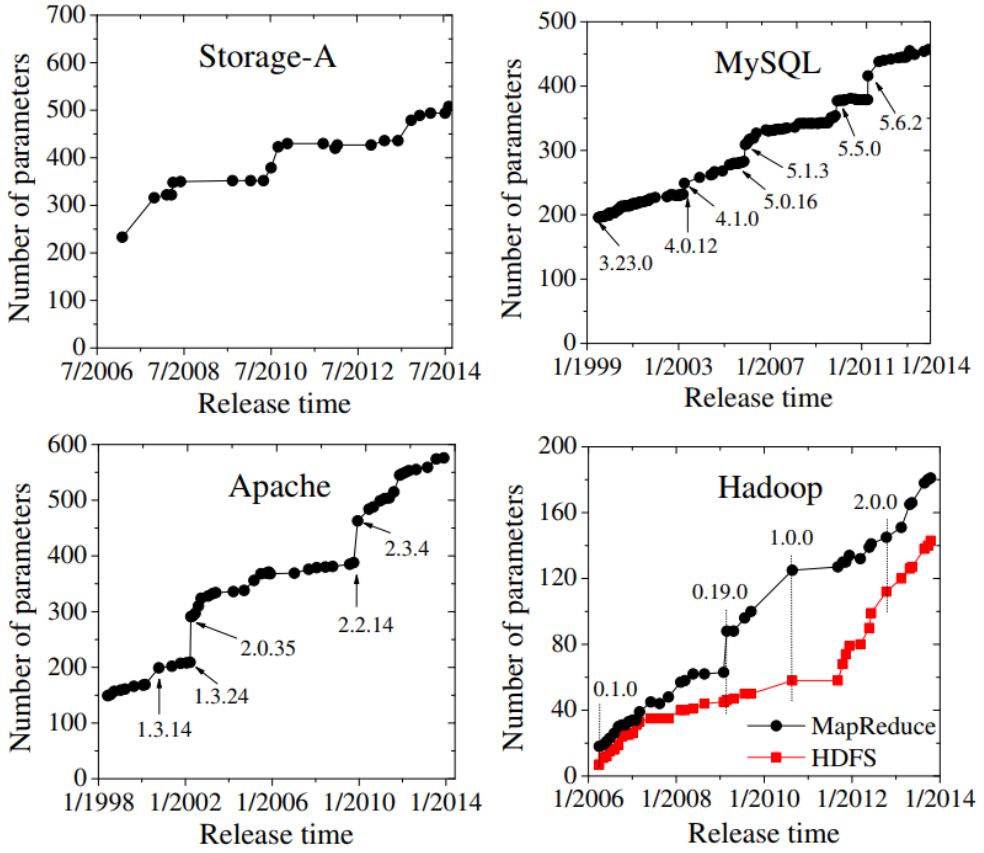




Predict, optimize, debug, and explain non-functional properties

Practical Relevance

Configuration complexity: [1] Xu et al. FSE'15: Developers and users are overwhelmed with configuration options



Parameter:	optimizer_prune_level (Boolean) /*MySQL*/
Desc.:	Controls the heuristics applied during query optimization to prune less-promising partial plans from the optimizer search space.
Values:	0 or 1
Usage:	No user set the parameter in our dataset.

(a) **Empirical, heuristic usages**

Parameter:	key_cache_block_size (Numeric) /*MySQL*/
Desc.:	The size in bytes of blocks in the key cache.
Values:	[512, 16384]
Usage:	All the users stay with the default value 1024 in our dataset.

Substantial increase in configurability

Unused optimization (up to 80% of options ignored)

Why Should We Care?

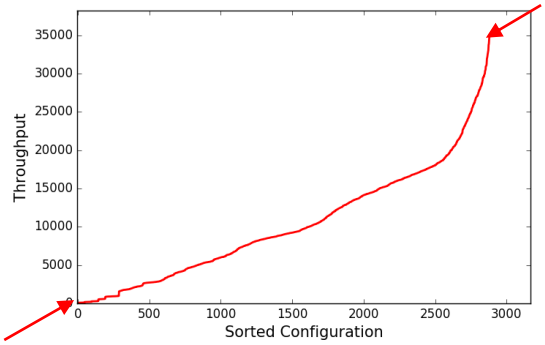
Outdated default configurations: [2] Van Aken et al. ICMD'17: Default configuration assumes 160MB RAM

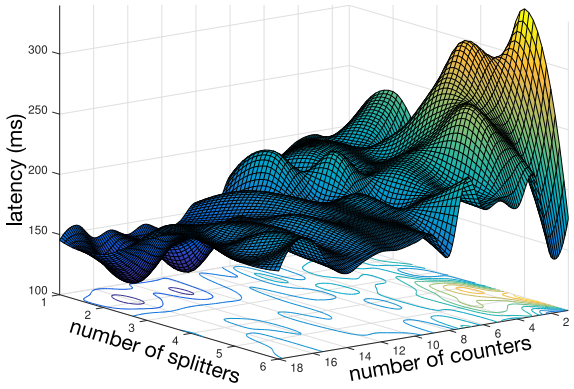
Non-optimal default configurations: [4] Herodotuo et al. CIDSR'11: Default configuration results in worst-case execution time

Non-optimal default configurations: [3] Jamshidi et al., MASCOTS'16: Changing configuration is key to tailor the system to the use case

Best

Worst





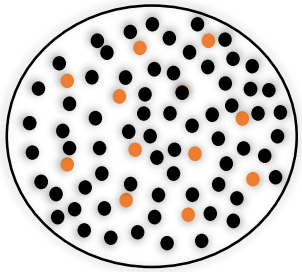
Best configuration is 480 times better than **Worst** configuration

Only by tweaking 2 options out of 200 in Apache Storm - observed **~100%** change in latency

Overview

Configuration space

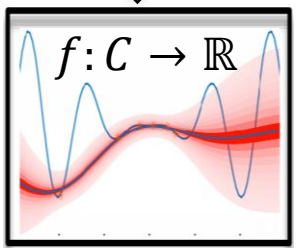
Size:~ $2^{\#options}$



(1) Sampling

Goal: Select a minimal, representative set of configurations
Challenges: Size of configuration space, constraints, interactions
Key domains: Combinatorial testing, artificial intelligence, search-based software engineering, design of experiments

Performance model



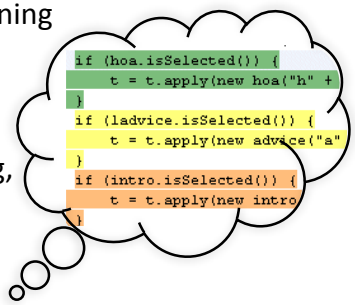
(2) Learning

Goal: Learn a model accurately describing performance of all configurations
Challenges: Dimensionality of the learning problem, interactions
Key domains: machine learning, statistics

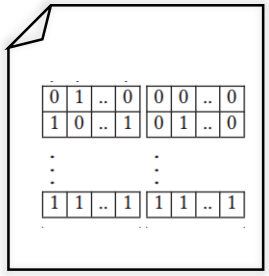
Goal

(4) Analysis

Goal: Explainable machine learning with white-box models
Challenges: Trade-off between explainability and accuracy
Key domains: machine learning, software analysis, testing



System understanding



Optimal configuration(s)

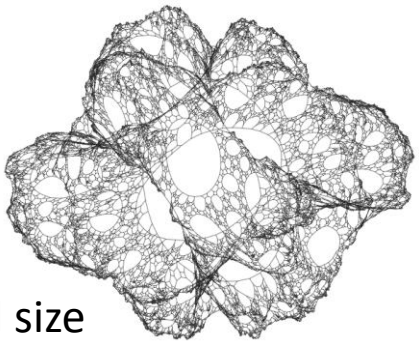
(3) Optimization

Goal: Finding optimal configurations in a single or multi-objective scenario
Challenges: Size of configuration space, constraints, interactions, lack of ground truth
Key domains: search-based software engineering, meta-heuristics, machine learning, artificial intelligence, mathematical optimization

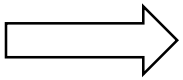
Which configurations to select for learning?

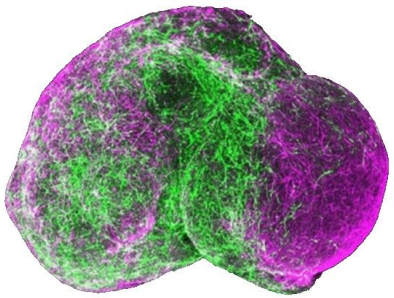
Sampling – Overview

Challenges:

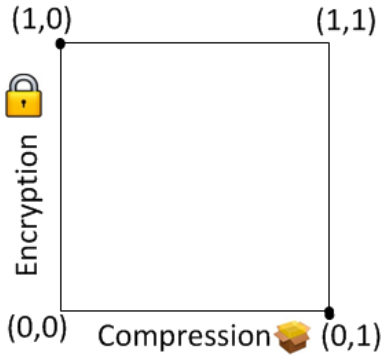


Exponential size configuration space

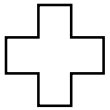


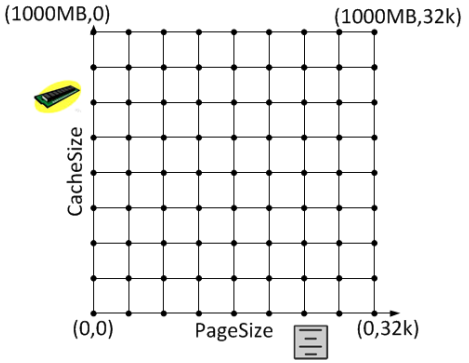


Find only relevant configurations for measurement



Binary configuration options





Numeric configuration options

Random Sampling

Or how to obtain randomness in the presence of **constraints**?

Trivial approach: Enumerate all configurations and randomly draw one

Not scalable

Easy to implement
True randomness

[12] Temple et al. TR'17; [13] Guo et al. ASE'13; [14] Nair et al. FSE'15; [15] Zhang et al. ASE'15;

SAT approach: Manipulate a SAT/CSP solver:

No guaranteed uniformity
Limited scalability

Easy to implement
Better distribution

[5] Henard et al. ICSE'15: Randomly permute constraint and literal order and phase selection (order true - false)
[17] Siegmund et al. FSE'17: Specify distribution of config. as constraints

Uniform Sampling of SAT Solutions for Configurable Systems: Are We There Yet?

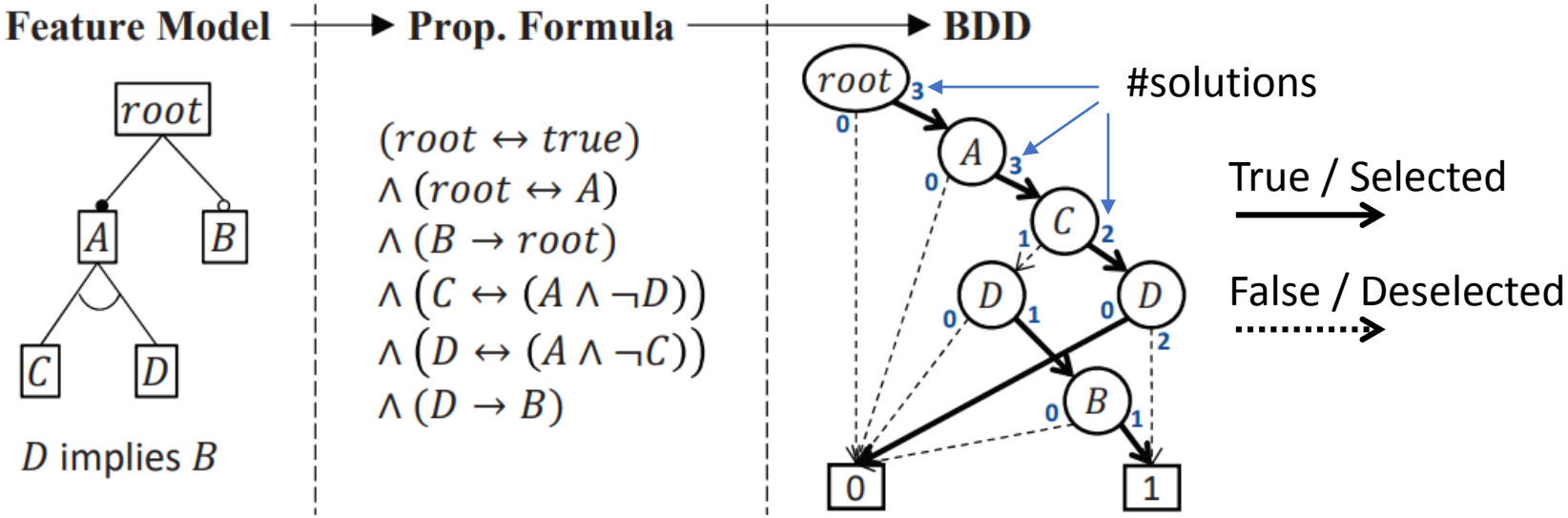
Quentin Plazar*, Mathieu Acher*, Gilles Perrouin†, Xavier Devroey‡ and Maxime Cordy§
*Univ Rennes, Inria, CNRS, IRISA, Rennes, France. Emails: quentin.plazar@gmail.com, mathieu.acher@irisa.fr
†PReCISE/NaDI, Faculty of Computer Science, University of Namur, Namur, Belgium. Email: gilles.perrouin@unamur.be
‡Delft University of Technology, Delft, The Netherlands. Email: x.d.m.devroey@tudelft.nl
§SnT, University of Luxembourg, Luxembourg, Luxembourg. Email: maxime.cordy@uni.lu

Still locally clustered solutions; no uniform sampling.

Beyond SE: Tailored algorithms: [7] Chakraborty et al. AAAI'14: Hash the configuration space
[8] Gogate and Dechter CP'06 and [9] Dechter et al. AAAI'02: Consider CSP output as probability distribution

Random Sampling with Counting BDD

Create a counting BDD to enumerate all configurations: [6] Oh et al. FSE'17



Example: Path(1,1,1,0,-) -> root=1, A=1, C=1, D=0 -> 1 -> True

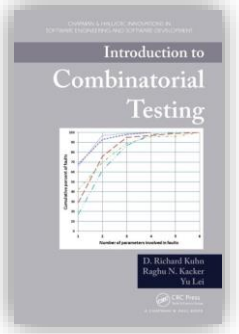
Approach: Root label = all valid solutions -> random number k -> convert k to configuration by traversal

BDD creation can be expensive

Scales up to 2,000 options
True randomness

Sampling with Coverage I

Survey: [10] Medeiros et al. ICSE'16

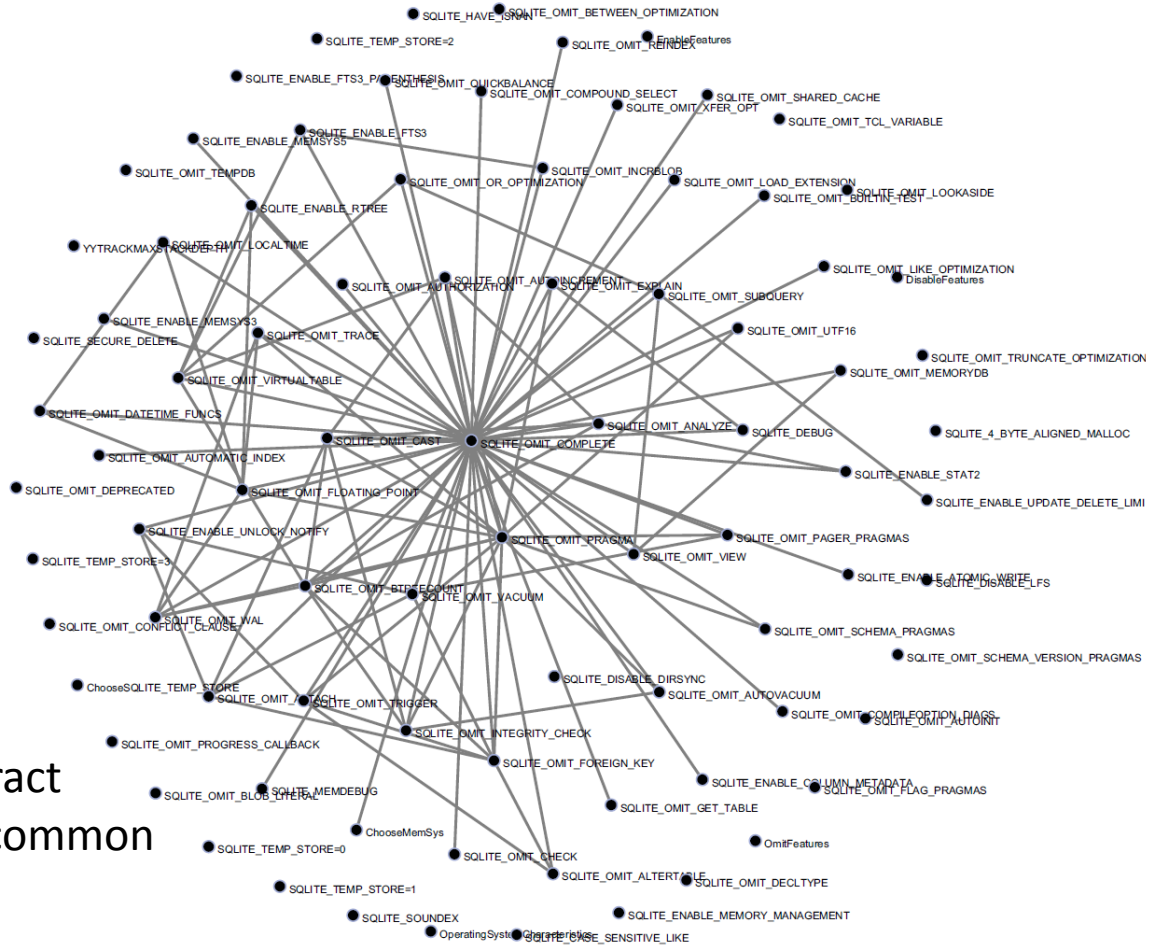


Kuhn et al.:
[11] Henard et al. TSE'14
[18] Cohen et al. TSE'08
[19] Johansen et al. SPLC'12

Interaction coverage: t-wise, (e.g., 2-wise = pair-wise)

[20] Siegmund et al. SPLC'11

[21] Siegmund et al. ICSE'12



Insights:

Many options do not interact

2-wise interactions most common

Hot-spot options

Sampling with Coverage II

Option coverage: Cover all options either by minimizing or maximizing interactions

```
#ifdef A
  // code 1
#endif

#ifdef B
  // code 2
#else
  // code 3
#endif

#ifdef C
  // code 4
#endif
```

pair-wise	
config-1:	!A !B C
config-2:	!A B !C
config-3:	A !B !C
config-4:	A B C

one-enabled	
config-1:	A !B !C
config-2:	!A B !C
config-3:	!A !B C

one-disabled	
config-1:	!A B C
config-2:	A !B C
config-3:	A B !C

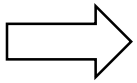
most-enabled-disabled	
config-1:	A B C
config-2:	!A !B !C

statement-coverage	
config-1:	A B C
config-2:	A !B C

Leave-one-out /one disabled sampling: [10] Medeiros et al. ICSE'16
Option-wise sampling: [20,24] Siegmund et al. SPLC'11, IST'13
Negative option-wise sampling: [22] Siegmund et al. FSE'15

Option-frequency sampling: Cover all options equally by Sakar et al. ASE'15 [23]

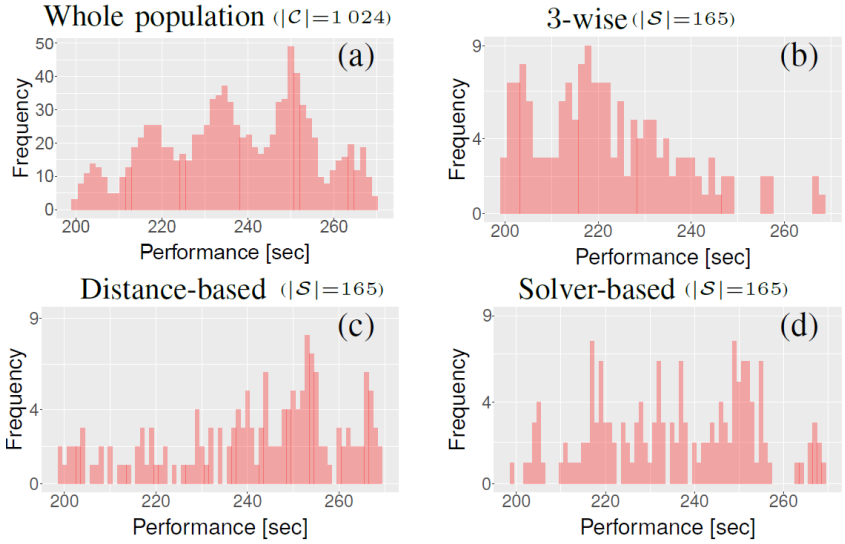
	x_1	x_2	x_3	..	x_i	..	..	x_N
1	0	1	1	0	0	0	1	1
2	0	0	1	1	1	0	0	0
3	1	1	0	1	0	1	1	1
4	0	1	0	1	0	1	0	0
5	1	1	0	0	0	1	0	1
6	0	0	0	1	1	1	0	0
7	1	1	0	1	0	0	0	1
8	1	0	0	0	1	0	0	1



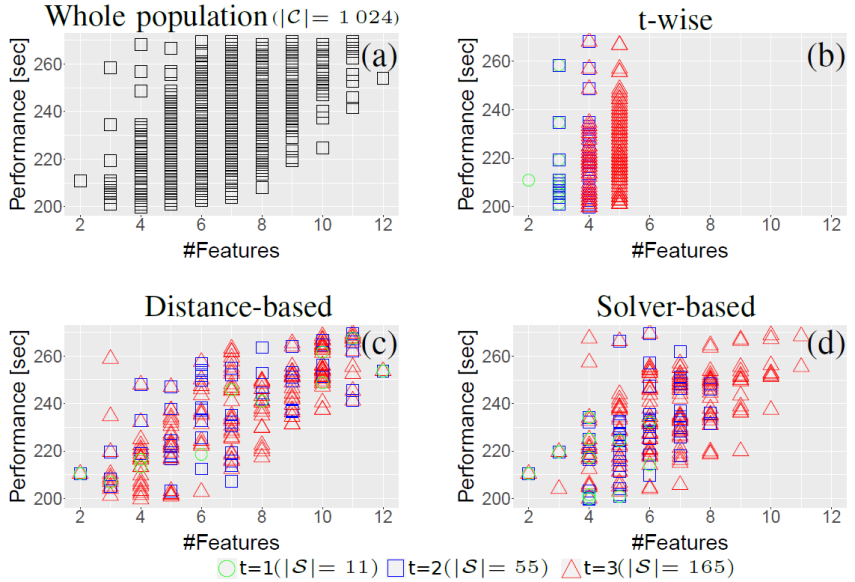
	x_1	x_2	x_3	..	x_i	..	..	x_N
selected	4	5	2	..	3	..	..	5
deselected	4	3	6	..	5	..	..	3

Distance-Based Sampling

Distance metric to select configurations with a given distribution by Kaltenecker et al. ICSE'19 [55]



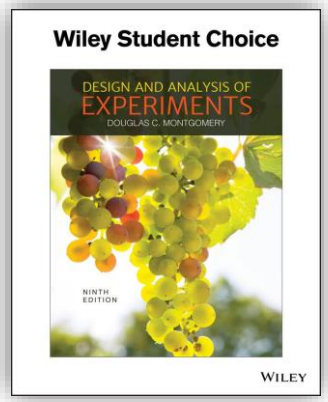
Performance distribution



Distance distribution of configurations

	Coverage-based			Solver-based			Randomized solver-based			Distance-based			Diversified distance-based			Random		
	$t=1$	$t=2$	$t=3$	$t=1$	$t=2$	$t=3$	$t=1$	$t=2$	$t=3$	$t=1$	$t=2$	$t=3$	$t=1$	$t=2$	$t=3$	$t=1$	$t=2$	$t=3$
7z	54.1 %	54.1 %	24.4 %	78.0 %	58.2 %	24.3 %	64.5 %	37.5 %	21.8 %	85.8 %	24.7 %	17.8 %	78.5 %	**19.6 %**	17.3 %	58.2 %	15.1 %	9.9 %
BDB-C	69.7 %	48.7 %	22.0 %	69.5 %	66.0 %	61.8 %	**65.0 %**	64.6 %	22.3 %	140.3 %	124.3 %	14.3 %	77.2 %	**45.6 %**	**13.6 %**	121.3 %	39.1 %	12.2 %
Dune	19.1 %	13.3 %	11.2 %	23.1 %	15.1 %	11.9 %	39.9 %	15.7 %	**10.9 %**	23.9 %	14.4 %	11.5 %	**17.3 %**	**12.7 %**	11.5 %	17.6 %	11.5 %	11.3 %
Hipacc	**26.2 %**	20.4 %	20.4 %	44.5 %	16.8 %	14.5 %	27.7 %	14.5 %	14.2 %	29.0 %	18.8 %	15.1 %	33.9 %	**13.9 %**	**13.4 %**	19.9 %	13.9 %	13.4 %
JavaGC	44.7 %	32.1 %	23.5 %	57.6 %	64.5 %	36.7 %	**40.6 %**	37.9 %	32.1 %	78.8 %	47.8 %	22.4 %	58.9 %	**15.4 %**	**13.2 %**	55.8 %	13.9 %	12.3 %
lrzip	**27.2 %**	**16.7 %**	11.3 %	51.9 %	26.4 %	24.3 %	42.2 %	20.8 %	16.3 %	122.9 %	33.9 %	17.3 %	122.2 %	26.7 %	14.3 %	62.7 %	18.3 %	15.6 %
LLVM	6.2 %	6.2 %	5.8 %	8.9 %	5.5 %	5.2 %	5.6 %	5.2 %	5.4 %	5.8 %	**5.2 %**	5.3 %	6.5 %	6.1 %	5.2 %	5.6 %	5.2 %	5.2 %
Polly	18.6 %	12.7 %	**7.4 %**	20.5 %	18.2 %	14.9 %	20.1 %	15.2 %	14.9 %	23.8 %	15.1 %	16.8 %	27.8 %	16.5 %	13.7 %	25.1 %	13.0 %	10.3 %
VP9	104.2 %	**36.5 %**	36.5 %	183.4 %	109.1 %	48.0 %	135.7 %	261.3 %	94.5 %	355.0 %	114.7 %	53.0 %	**80.0 %**	42.1 %	**32.0 %**	80.6 %	27.2 %	23.3 %
x264	16.5 %	**9.3 %**	**9.0 %**	25.4 %	39.7 %	42.6 %	18.2 %	25.8 %	32.9 %	13.6 %	10.1 %	9.8 %	**13.5 %**	9.4 %	9.1 %	13.5 %	9.2 %	9.1 %
Mean	**38.6 %**	25.0 %	17.1 %	56.3 %	42.0 %	28.4 %	45.9 %	49.8 %	26.5 %	87.9 %	40.9 %	18.3 %	51.6 %	**20.8 %**	**14.3 %**	46.0 %	16.6 %	12.3 %

Sampling Numeric Options



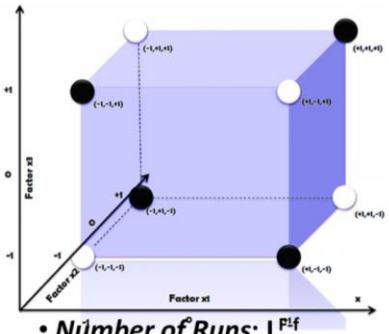
Identification of Main Effects with all other interactions negligible

PLACKETTE BURMAN

	X_1	X_2	X_3	X_4	X_5	X_6	X_7	X_8	X_9	X_{10}	X_{11}
1	+1	+1	+1	+1	+1	+1	+1	+1	+1	+1	+1
2	-1	+1	-1	+1	+1	+1	-1	-1	-1	+1	-1
3	-1	-1	+1	-1	+1	+1	+1	-1	-1	-1	+1
4	+1	-1	-1	+1	-1	+1	+1	+1	-1	-1	-1
5	-1	+1	-1	-1	+1	-1	+1	+1	+1	-1	-1
6	-1	-1	+1	-1	-1	+1	-1	+1	+1	+1	-1
7	-1	-1	-1	+1	-1	-1	+1	-1	+1	+1	+1
8	+1	-1	-1	-1	+1	-1	-1	+1	-1	+1	+1
9	+1	+1	-1	-1	-1	+1	-1	-1	+1	-1	+1
10	+1	+1	+1	-1	-1	-1	+1	-1	-1	+1	-1
11	-1	+1	+1	+1	-1	-1	-1	+1	-1	-1	+1
12	+1	-1	+1	+1	+1	-1	-1	-1	+1	-1	-1

Identification of Main Effects confounded with 2 way interactions when resources are limited

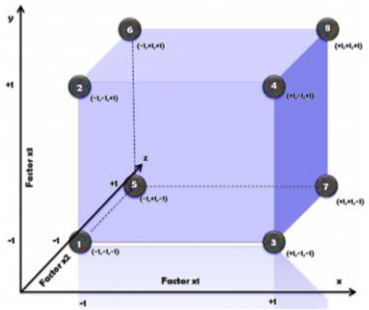
FRACTIONAL FACTORIAL



• *Number of Runs*: L^{F-f}

Identification of Main Effects **WITH** 2 way interactions

FULL FACTORIAL

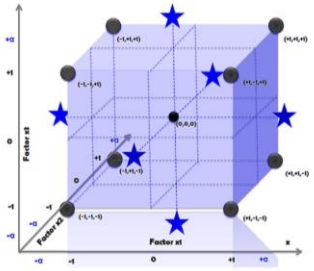


• *Number of Runs*: L^{F} .

- The goal is **OPTIMIZATION** of critical factors
- **Emphasis is on the fitted surface representing true behavior**
- Detect non-linear **significant curvature in response surface** to investigate **full quadratic relationship**

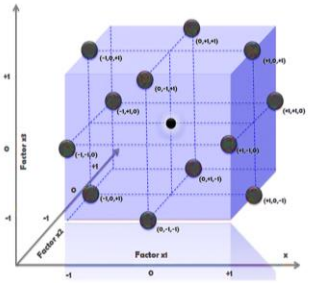
Response Surface

CENTRAL COMPOSITE



- ***Levels:*** **5** [''- α', `-1' '0' and `+1',' + α']
- ***No of Runs:*** $2^{fp} + 2SP + CP$

BOX BEHNKEN



- ***Levels:*** **3** levels per factor
- ***Design Points:*** at the "**mid" points of edges** of the process space & center

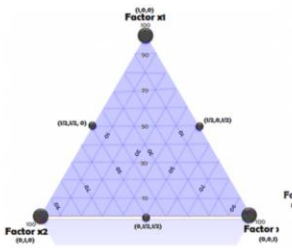
- **Factors** are **COMPONENTS** of a **MIXTURE** &
- Components must **TOTAL TO A CONSTANT** i.e. **1 (100%).**
- **Response** is a **function of proportion of mixture components.**

Mixture

All **components** have the **same range**

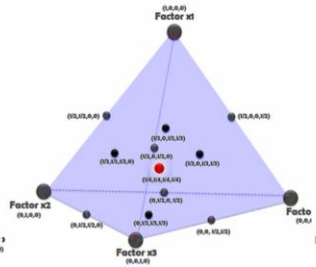
Upper and/or lower bound constraints.

SIMPLEX LATTICE



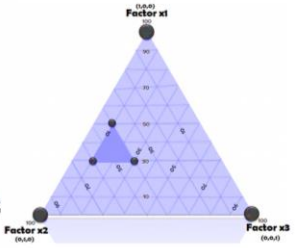
Number of design points *in the {q, m } simplex-lattice* ***(q+m-1)! /(m!(q-1)!).***

SIMPLEX CENTROID



In the q-component, the ***number of distinct points is*** $2^q - 1$.

D- OPTIMAL CONSTRAINED



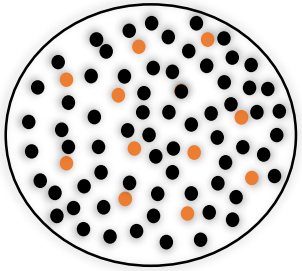
Number of Design points

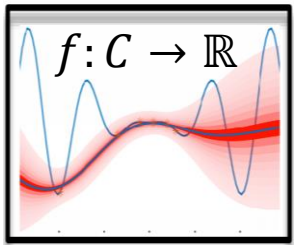
$x_i = L_i + (1 - L)\, x_i$*

L = sum of all lower bounds.

How do I learn a performance model?

Learning Performance Models

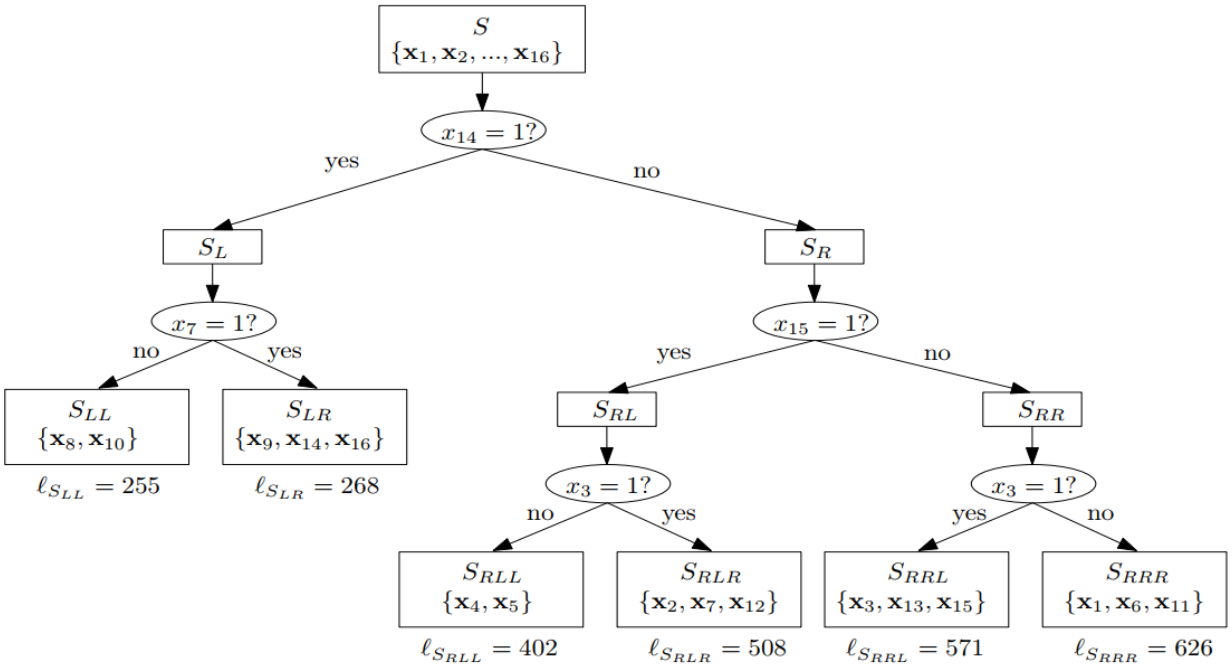




Predict any configuration
Find (near-)optimal configuration
Find influencing options/interactions

Accurate prediction: Using classification and regression trees (CART)

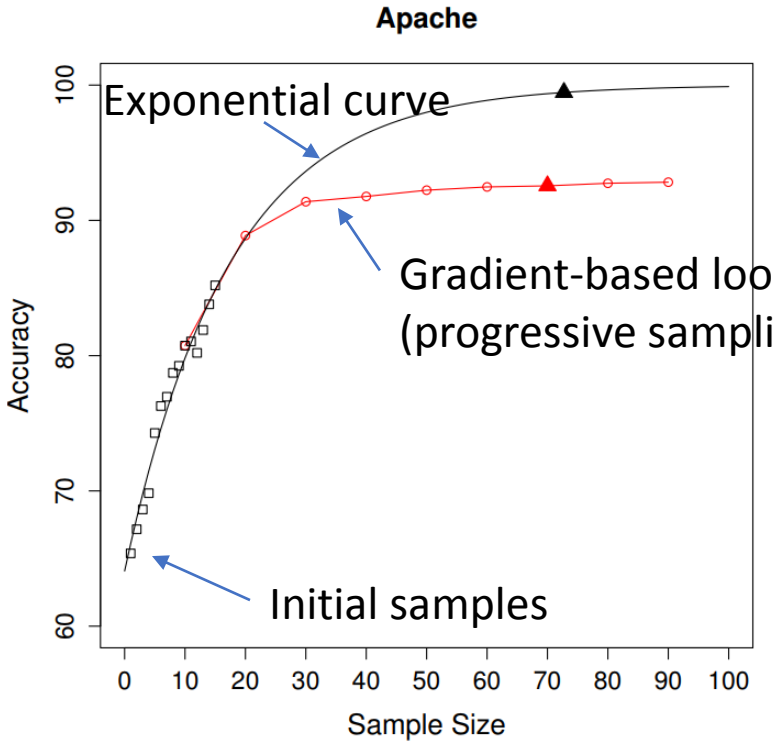
[13] Guo et al. ASE'13:



Learning Performance Models I

Accurate prediction: CART + feature-frequency sampling + early abortion

[23] Sakar et al. ASE'15: Plot #samples with accuracy and fit a function telling when to abort

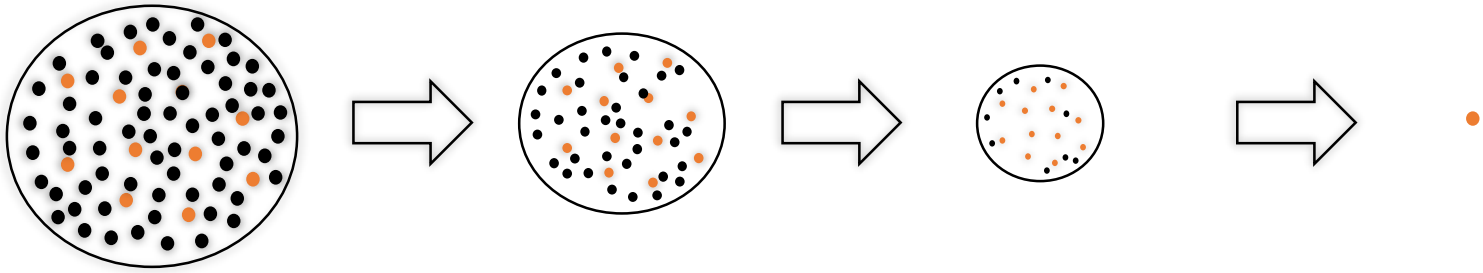


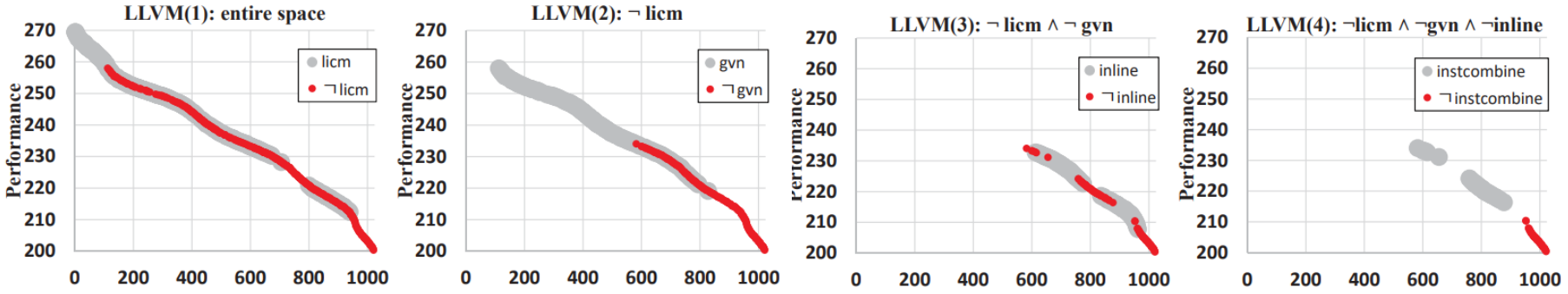
Name	Equation	Optimal Sample Size
Logarithmic	$err(n) = a + b.log(n)$	$n^* = -(R \cdot \lvert S \rvert \cdot b)/2$
Weiss and Tian	$err(n) = a + bn/(n+1)$	$n^* = \sqrt{(-R \cdot \lvert S \rvert \cdot b)/2}$
Power Law	$err(n) = an^b$	$n^* = (\frac{-2}{R \cdot \lvert S \rvert \cdot a \cdot b})^{\frac{1}{b-1}}$
Exponential	$err(n) = ab^n$	$n^* = \log_b\left(\frac{-2}{R \cdot \lvert S \rvert \cdot a \cdot \ln b}\right)$

State-of-the-art approach for accuracy-measurement tradeoff

Learning Performance Models II

Finding near-optimal configurations: [6] Oh et al. FSE'17: True random sampling + select best in sample set + infer good/bad options + shrink configuration space accordingly + repeat

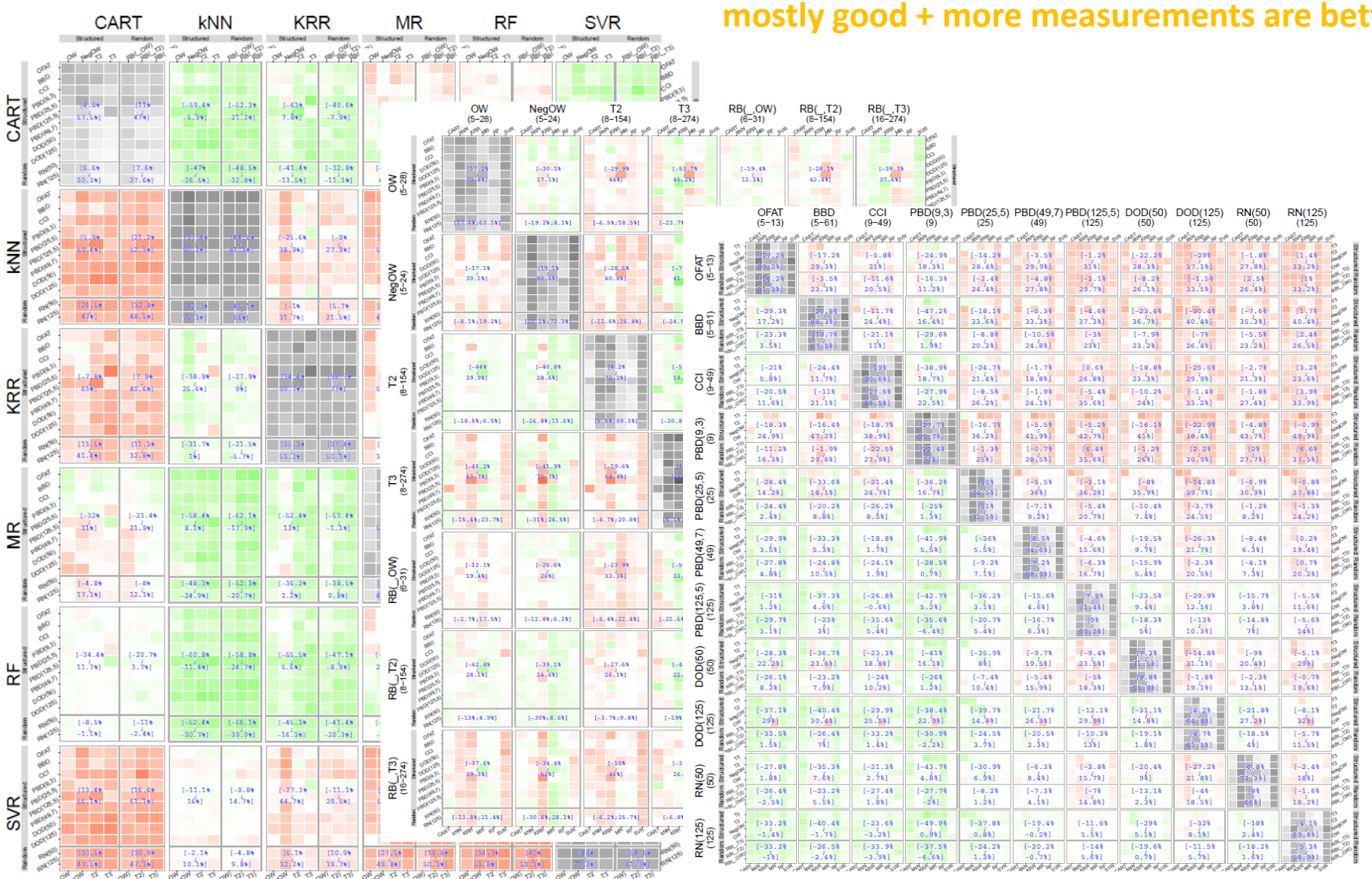




State-of-the-art approach for finding the near-optimal configuration with minimal #measurements

Which Combination of Learning and Sampling?

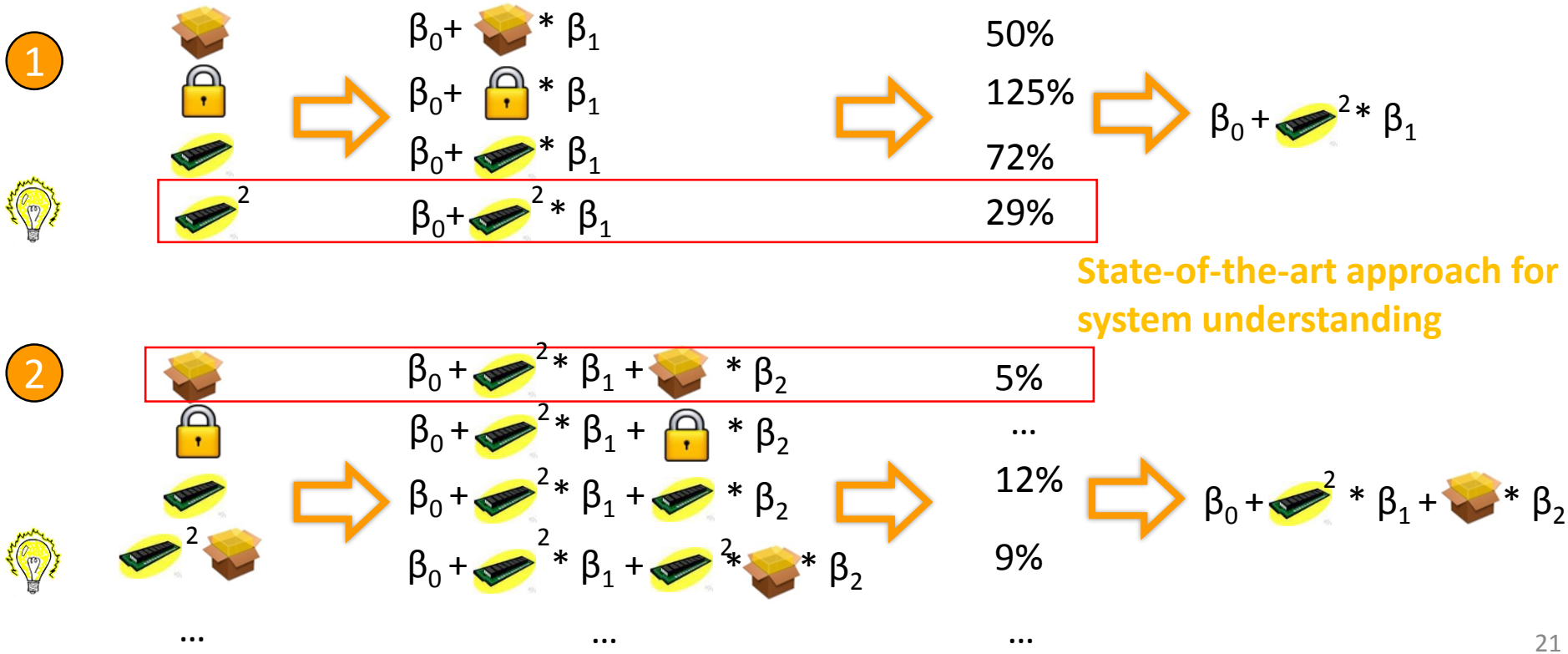
No silver bullet: not a single combination is superior in all cases, but random forest is mostly good + more measurements are better



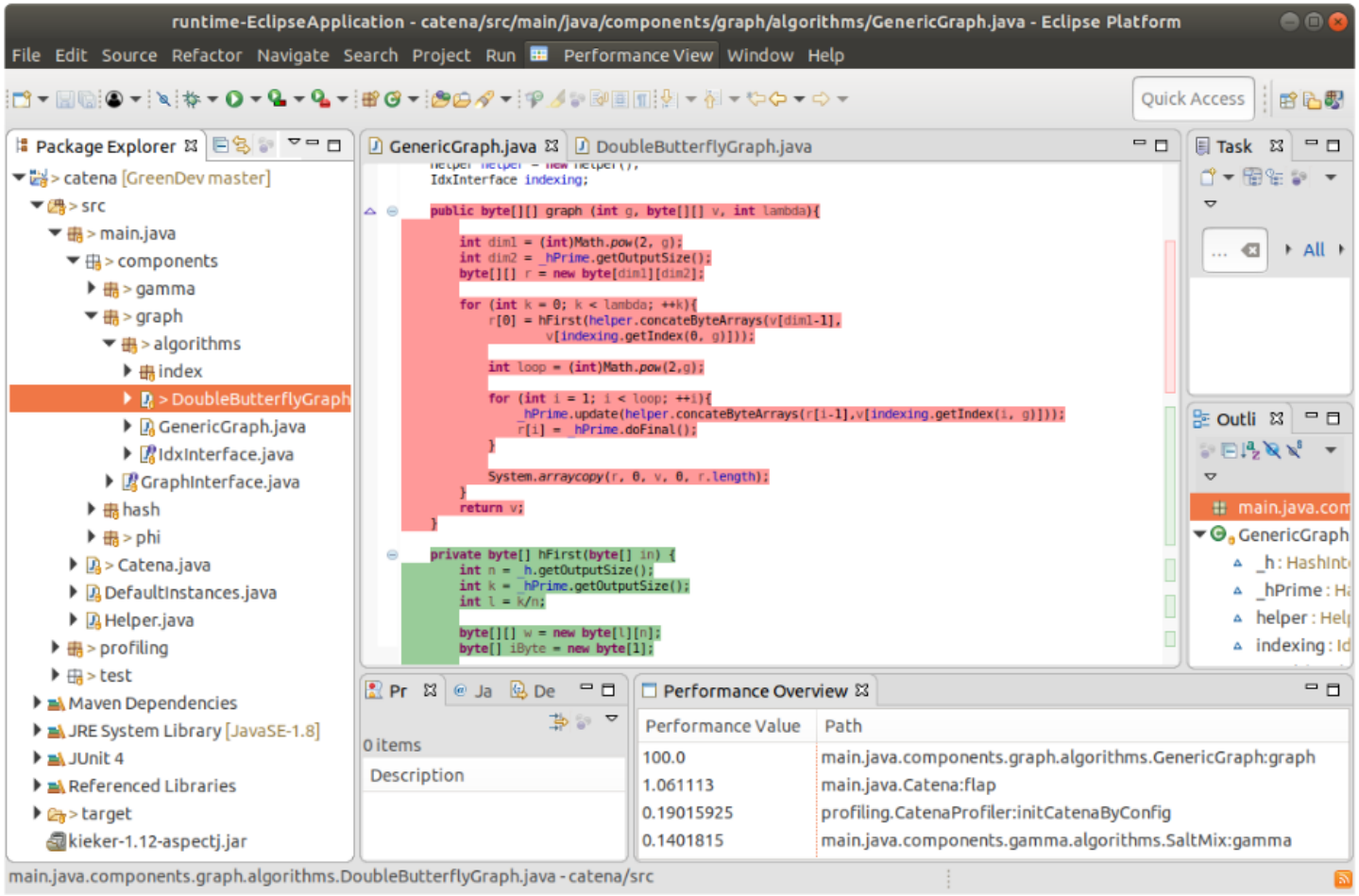
Which option/code is responsible for this performance?

Responsible Options: Multi-Variable Regression

System understanding: [22] Siegmund et al. FSE'15: Find influencing options and interactions via step-wise construction of performance model using multivariable regression

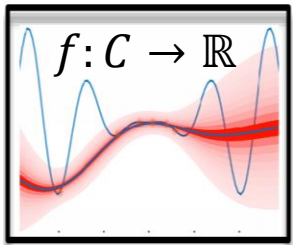


Responsible Code: Profiling

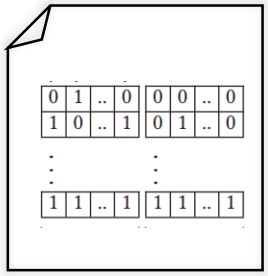


How do I find the optimal configuration?

Optimization Overview

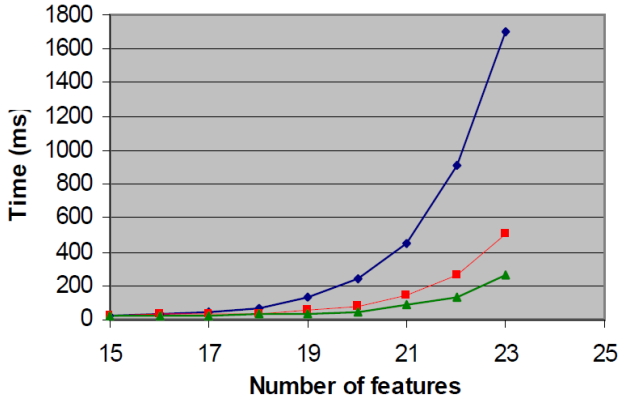


Surrogate model



Single-objective optimization
Multi-/Many-objective optimization
Partial configuration support

[33] Benavides et al. CAiSE'05 : Translating to constraint satisfaction problem
[16] Siegmund et al. SQJ'12: Similar as [33] + qualitative constraints



Problem: Exponential solving time (NP-hard); proved in:

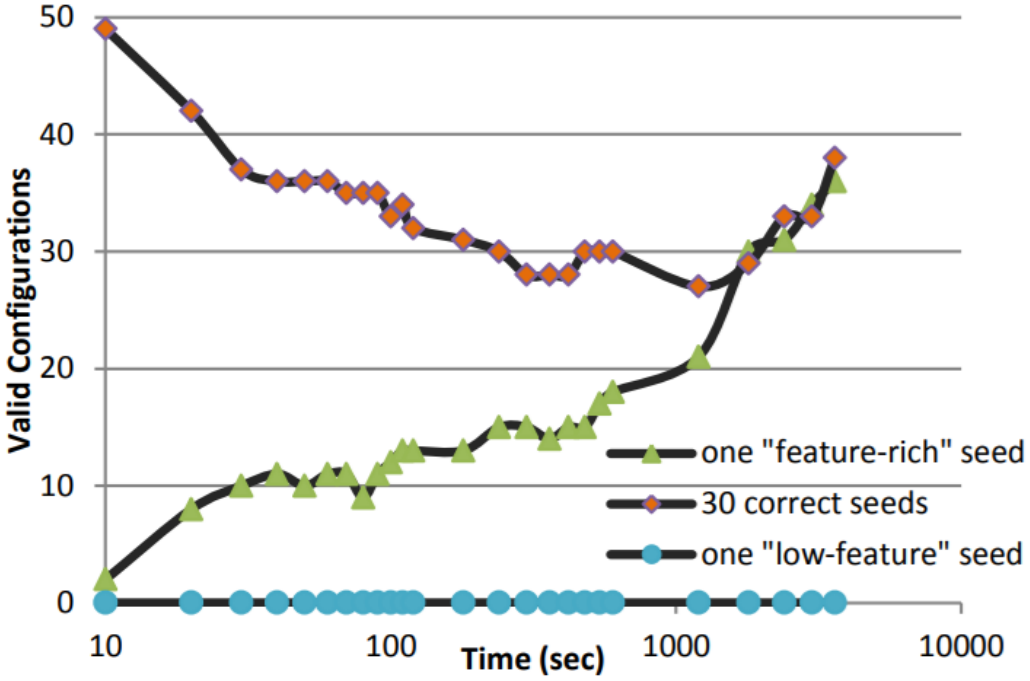
[24] White et al. JSS'09: Translating to knapsack problem via filtered cartesian flattening

Solution: Non-exact method, such as meta-heuristics, with main focus on how to handle constraints

Meta-Heuristic Based Optimization

Fix invalid configurations: [26] Guo et al. JSS'11: Genetic algorithm + search in invalid space + repair operation to return in valid configuration space

Encode constraints as additional objectives: [31,32] Sayyad et al. ICSE'13,ASE'13: Genetic algorithm (NSGA-II + IBEA) + improving fitness by reducing unsatisfied constraints

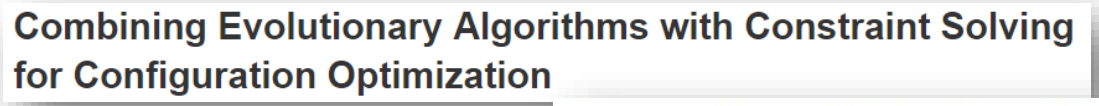


Scalability problems (30mins for 30 valid solutions based on 1 initial valid solution)

And many more…

[39] Tan et al. ISSTA'15

[40] Hierons et al. TOSEM'16



[41] Kai Shi ICSME'17

[42] Olaechea et al. SPLC'14

However, many approaches suffer unrealistic and too simplistic settings, and the results are not reproducible or do not hold for other settings.

How to obtain valid and reproducible results?

Know the Basics

A Few Useful Things to Know about Machine Learning

Pedro Domingos
Department of Computer Science and Engineering
University of Washington
Seattle, WA 98195-2350, U.S.A.
pedrod@cs.washington.edu

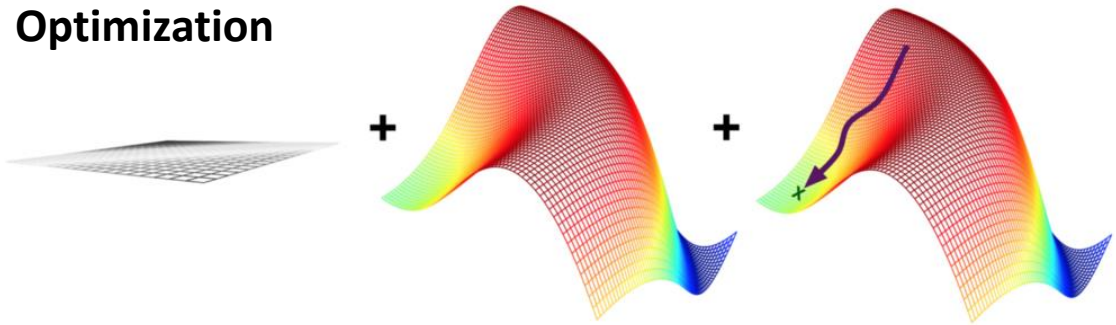
ABSTRACT
Machine learning algorithms can figure out how to perform important tasks by generalizing from examples. This is often feasible and cost-effective where manual programming is not. As more data becomes available, more ambitious problems can be tackled. As a result, machine learning is widely used in computer science and other fields. However, developing successful machine learning applications requires a substantial amount of "black art" that is hard to find in textbooks. This article summarizes twelve key lessons that machine learning researchers and practitioners have learned. These include pitfalls to avoid, important issues to focus on, and answers to common questions.

correct output y_t for future examples $\mathbf{x_t}$ (e.g., whether the spam filter correctly classifies previously unseen emails as spam or not spam).

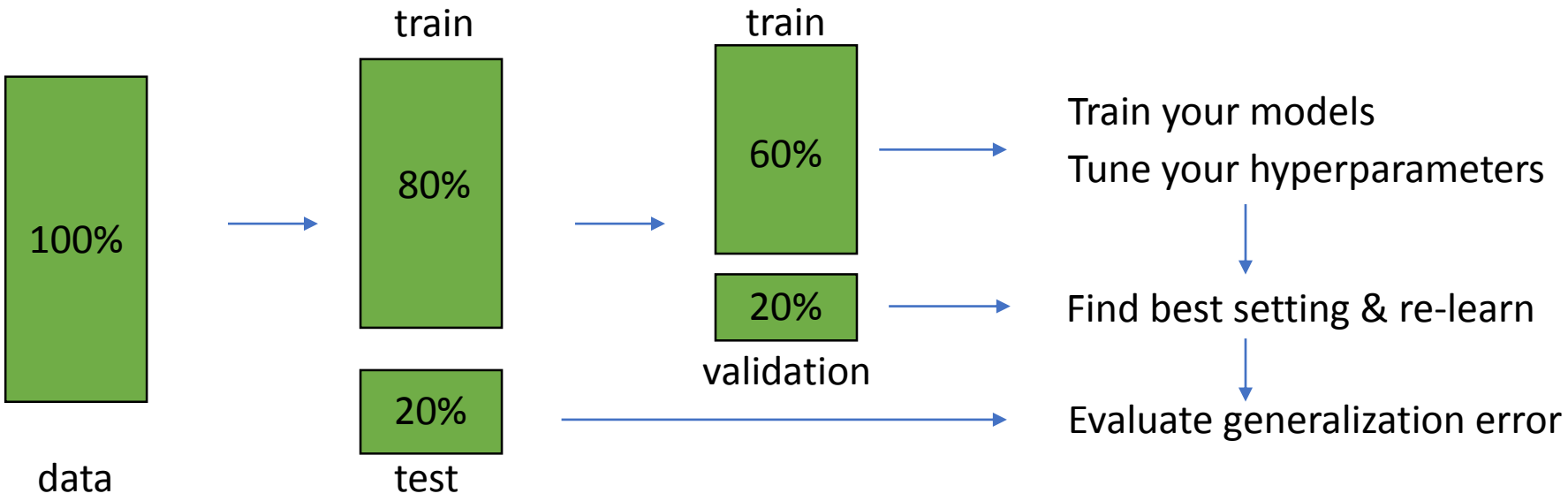
2. LEARNING = REPRESENTATION + EVALUATION + OPTIMIZATION

Suppose you have an application that you think machine learning might be good for. The first problem facing you is the bewildering variety of learning algorithms available. Which one to use? There are literally thousands available, and hundreds more are published each year. The key to not getting lost in this huge space is to realize that it consists of combinations of just three components. The components are:

Learning = Representation + Evaluation + Optimization

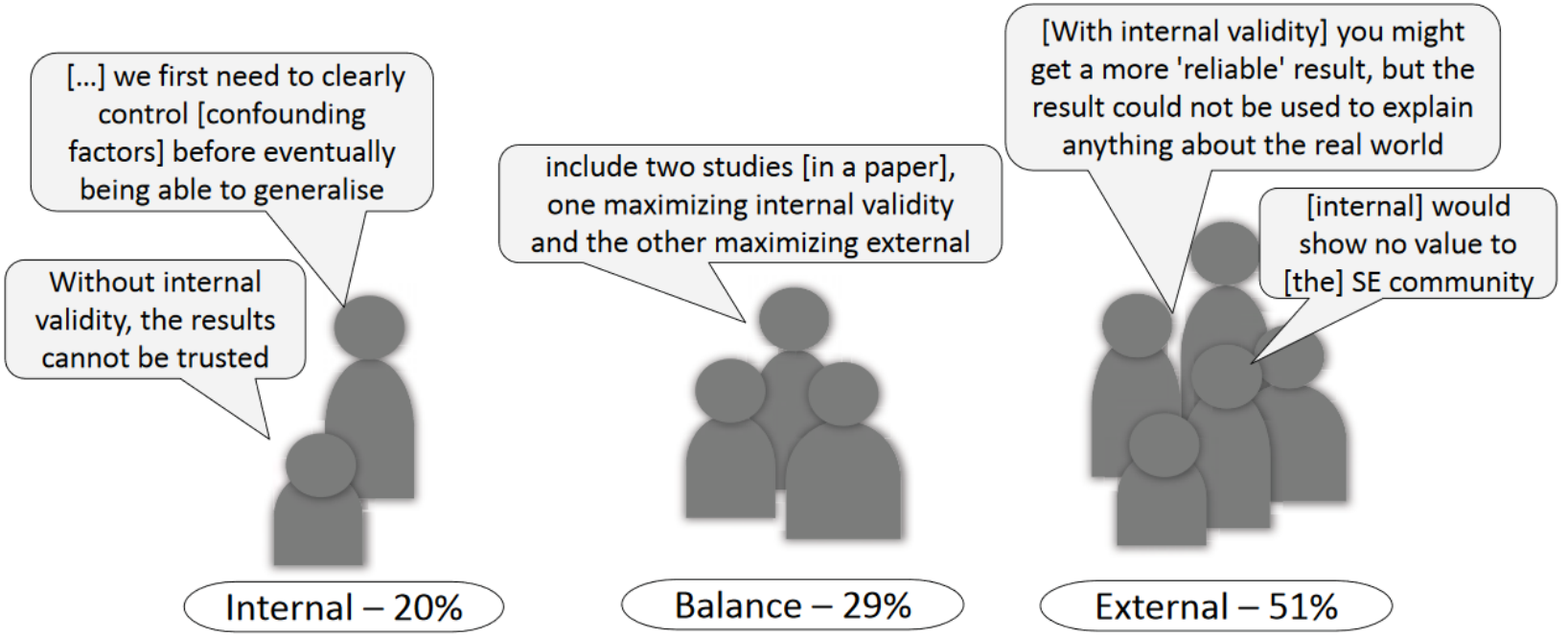


It's Generalization that Counts



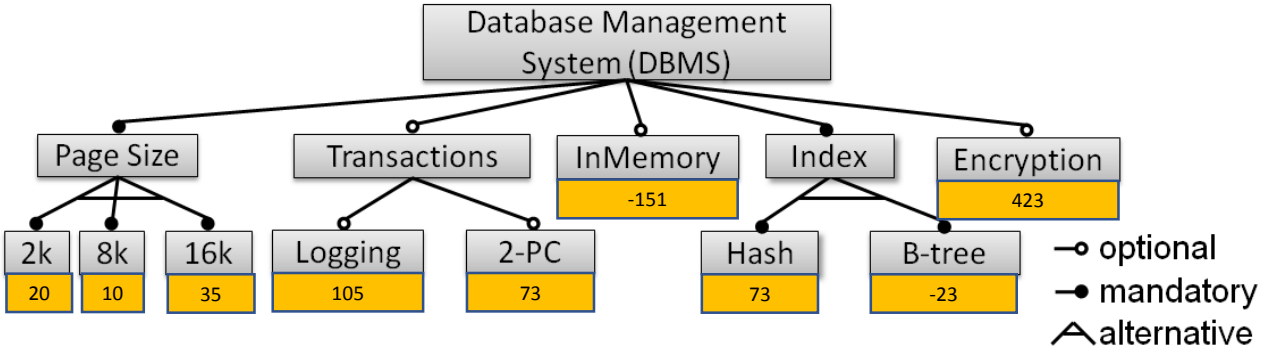
Optimize for Internal and External Validity

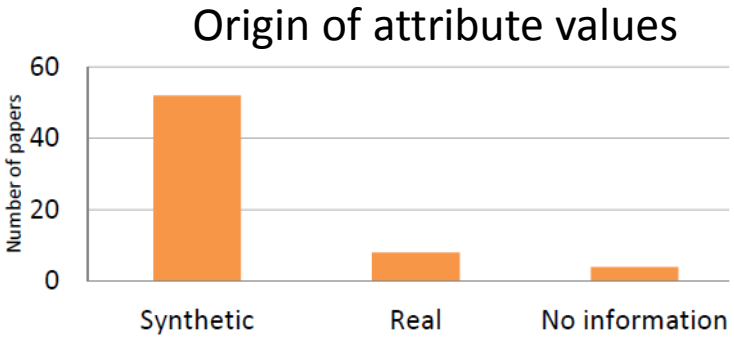
Viewpoints on validity: [26] Siegmund et al. ICSE'15: Program committee members disagree about the importance of internal and external validity + replication is needed, but not rewarded + community does not know how to treat empirical research

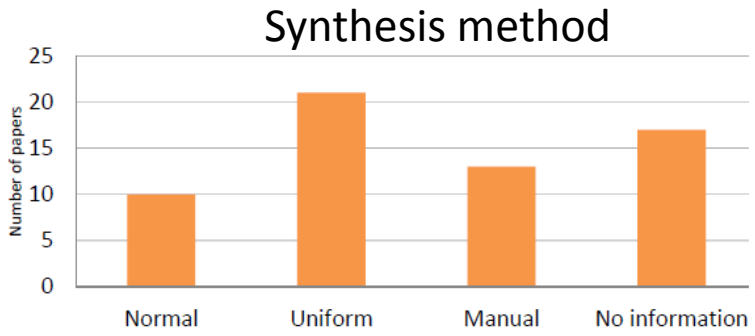


Internal: Realistic, Controlled Setting

Realistic attributed variability model: 17] Siegmund et al. FSE'17: Evaluating the state and complexity of used attributed variability models (AVM) in research + unrealistic settings used by nearly all papers + replication yields contradicting results + tool (Thor) to improve situation







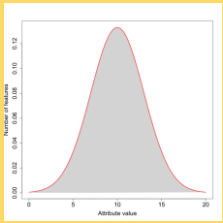
2,346 papers analyzed with 69 of them using AVM

Real attribute values
+ interactions

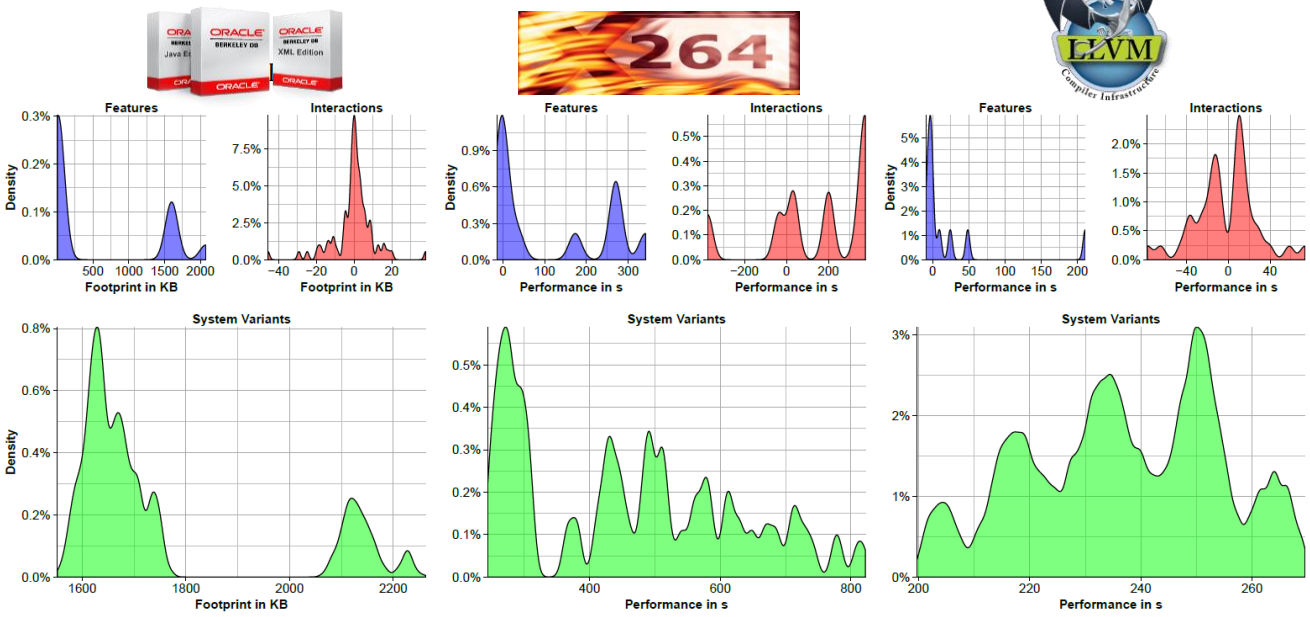
vs.

The Comfort Zone

No interactions +

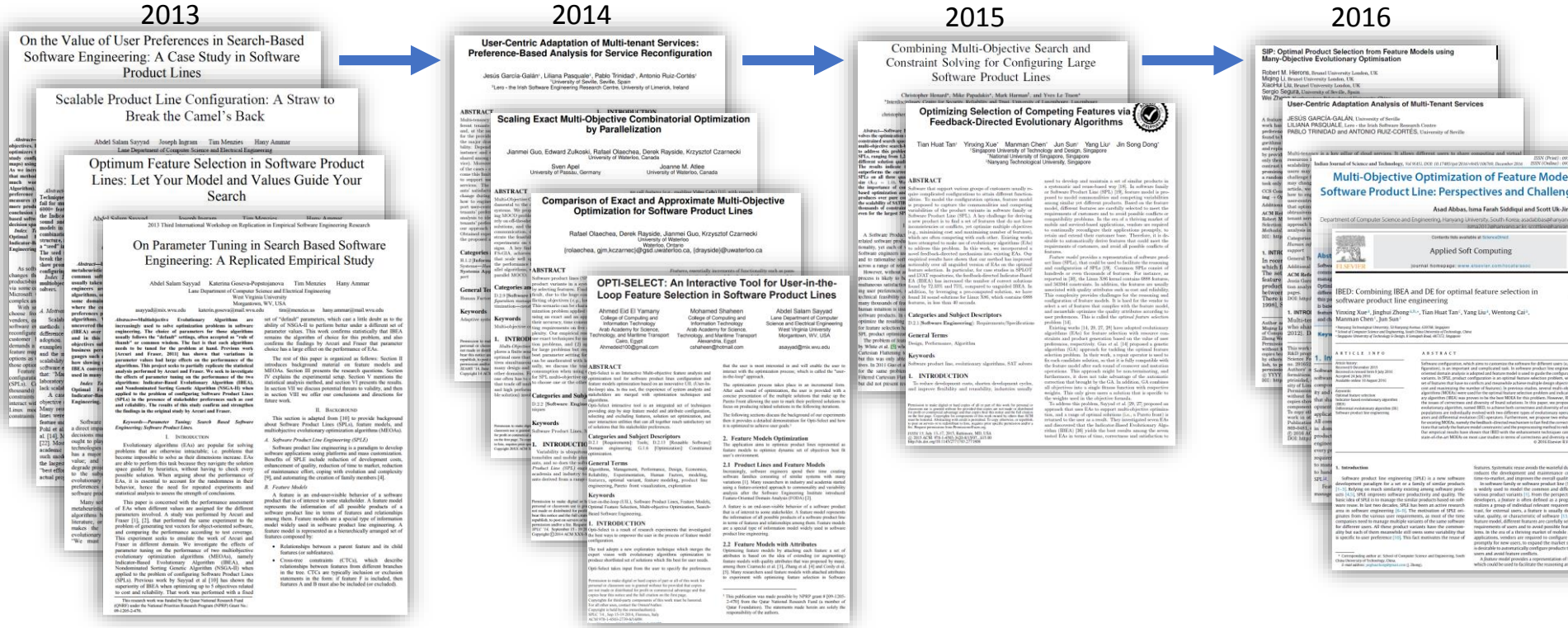


artificial value distributions



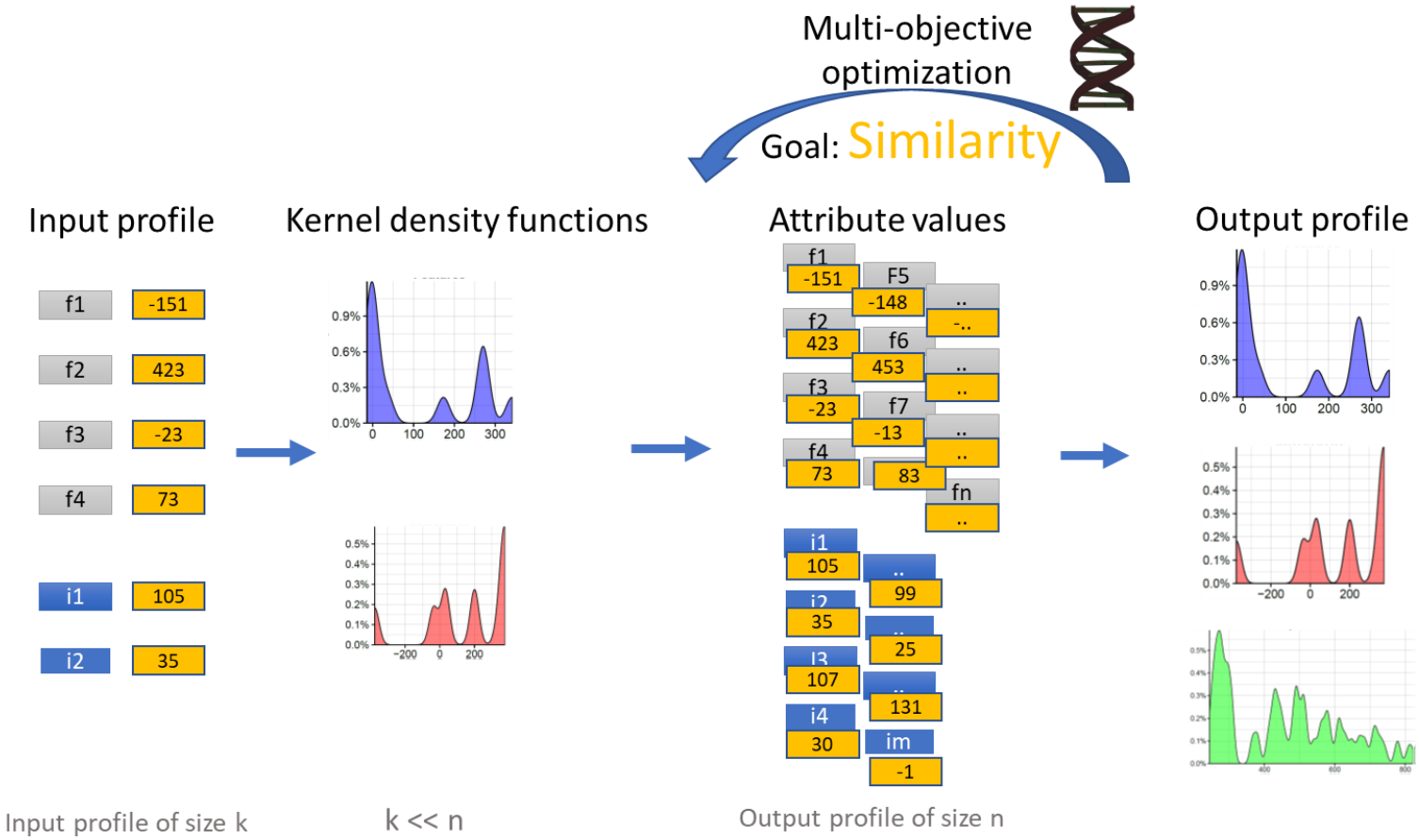
From 69 papers: 0 include interactions!

A Single learning setup for Many approaches



Replication-Extension of [31,32] shows: One Setting is Not Enough

Generating Realistic Settings



Thor, the accompanying tool

https://github.com/se-passau/thor-avm

What if my environment changes?

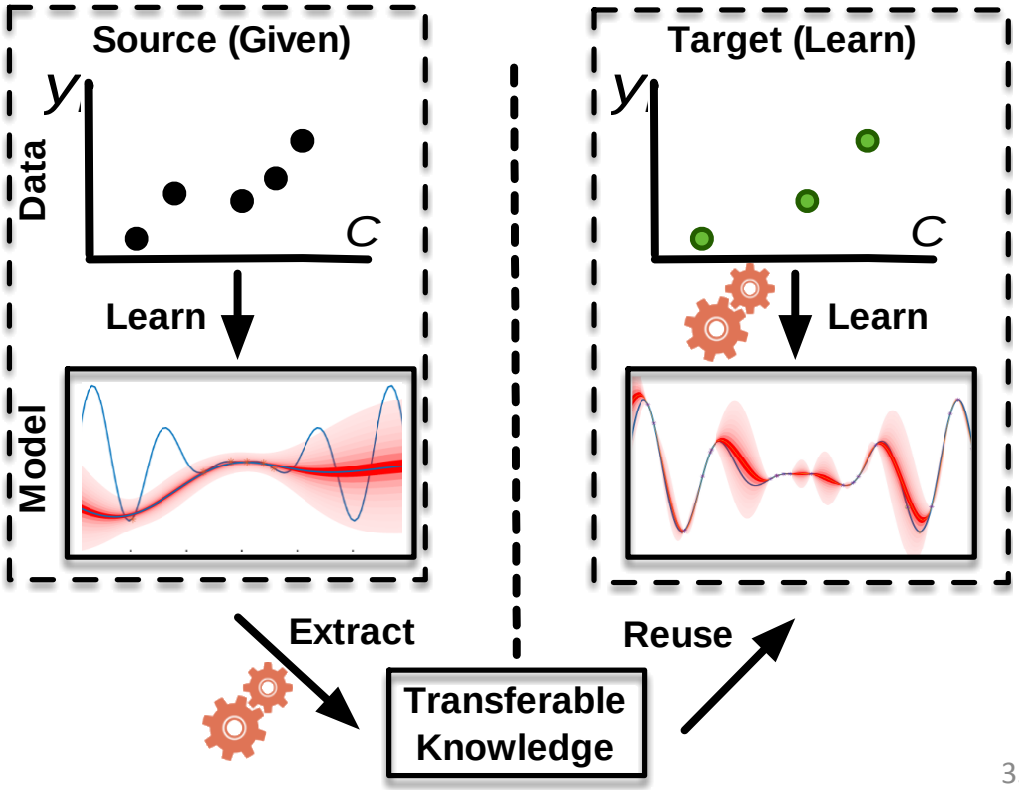
Transfer Learning I

So far, one performance model for one scenario/workload/hardware:

WHAAAT?

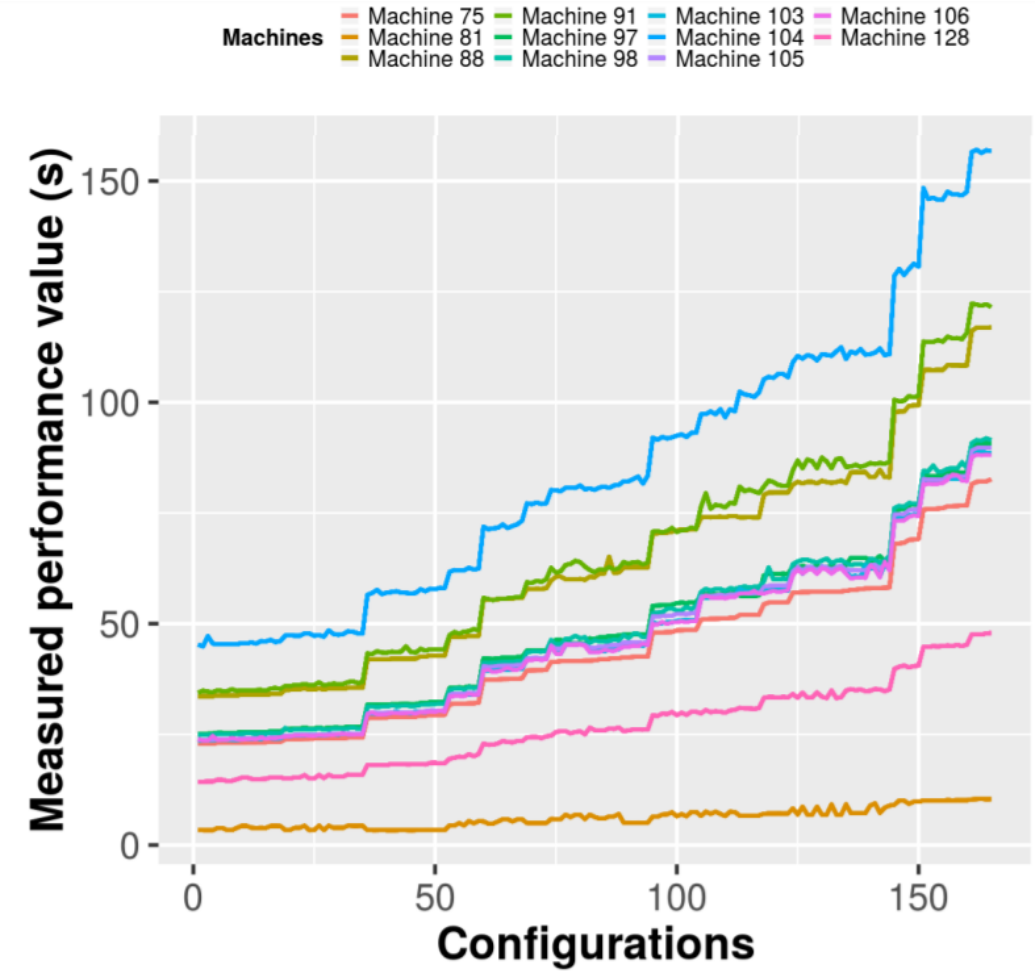
Environment change - >
New performance model

Transfer Learning



Transfer Learning II

Handle hardware changes: [43] Valov et al. ICPE'17: Adapt a learned performance model to a changed hardware using a linear function



Handles linear changes

Transfer Learning IV

Handle arbitrary changes: [45] Jamshidi et al. ASE'17: Empirical analysis about transferable knowledge of environmental changes (hardware, software version, workload)

TABLE II: Results indicate that there exist several forms of knowledge that can be transfered across environments and can be used in transfer learning.

		RQ1				RQ2				RQ3				RQ4			
		H1.1	H1.2	H1.3	H1.4	H2.1		H2.2		H3.1		H3.2		H4.1		H4.2	
$ec_6 : [h_1, w_1 \rightarrow w_2, v_1 \rightarrow v_2]$	L	-0.10	**0.72**	-0.05	0.35	0.04	5	6	1	3	**0.68**	**7**	**21**	7	0.31	**0.50**	**0.45**	**1**	**0.96**
$ec_7 : [h_1 \rightarrow h_2, w_1 \rightarrow w_4, v_2 \rightarrow v_1]$	VL	-0.10	6.95	0.14	0.41	0.15	6	4	2	2	**0.88**	**21**	**7**	7	-0.44	**0.47**	**0.50**	**1**	**0.97**
x264— Workload (#pictures/size): $w_1 : 8/2, w_2 : 32/11, w_3 : 128/44$; Version: $v_1 : r2389, v_2 : r2744, v_3 : r2744$																			
$ec_1 : [h_2 \rightarrow h_1, w_3, v_3]$	SM	**0.97**	1.00	**0.99**	**0.97**	**0.92**	9	10	8	0	0.86	21	33	18	1.00	0.49	0.49	1	1
$ec_2 : [h_2 \rightarrow h_1, w_1, v_3]$	S	**0.96**	0.02	**0.96**	**0.76**	**0.79**	9	9	8	0	0.94	36	27	24	1.00	0.49	0.49	1	1
$ec_3 : [h_1, w_1 \rightarrow w_2, v_3]$	M	0.65	**0.06**	0.63	0.53	0.58	9	11	8	1	**0.89**	27	33	22	**0.96**	0.49	0.49	1	1
$ec_4 : [h_1, w_1 \rightarrow w_3, v_3]$	M	0.67	**0.06**	0.64	0.53	0.56	9	10	7	1	**0.88**	27	33	20	**0.96**	0.49	0.49	1	1
$ec_3 : [h_2, w_1 \rightarrow w_2, v_1]$	S	**0.96**	1.27	**0.83**	0.40	0.35	2	3	1	0	1	9	9	7	0.99	N/A	N/A	N/A	N/A
$ec_4 : [h_2, w_3 \rightarrow w_4, v_1]$	M	0.50	**1.24**	0.43	0.17	0.43	1	1	0	0	1	4	2	2	**1.00**	N/A	N/A	N/A	N/A
$ec_5 : [h_1, w_1, v_1 \rightarrow v_2]$	M	**0.95**	1.00	0.79	0.24	0.29	2	4	1	0	1	12	11	7	0.99	N/A	N/A	N/A	N/A
$ec_6 : [h_1, w_2 \rightarrow w_1, v_1 \rightarrow v_2]$	L	0.51	**2.80**	0.44	0.25	0.30	3	4	1	1	0.31	7	11	6	**0.96**	N/A	N/A	N/A	N/A
$ec_7 : [h_2 \rightarrow h_1, w_2 \rightarrow w_1, v_1 \rightarrow v_2]$	VL	0.53	4.91	0.53	0.42	0.47	3	5	2	1	0.31	7	13	6	**0.97**	N/A	N/A	N/A	N/A
SaC— Workload: $w_1 : srad, w_2 : pfilter, w_3 : kmeans, w_4 : hotspot, w_5 : nw, w_6 : nbody100, w_7 : nbody150, w_8 : nbody750, w_9 : gc, w_{10} : cg$																			
$ec_8 : [h_1, w_3 \rightarrow w_4, v_1]$	L	0.68	**1.70**	0.56	0.00	**0.91**	14	13	9	1	**0.88**	57	67	36	0.34	0.11	0.14	0.05	0.67
$ec_9 : [h_1, w_3 \rightarrow w_5, v_1]$	VL	0.06	3.68	0.20	0.00	0.64	16	10	9	0	**0.90**	51	58	35	-0.52	0.11	0.21	0.06	-0.41
$ec_{10} : [h_1, w_4 \rightarrow w_5, v_1]$	L	0.70	4.85	0.76	0.00	**0.75**	12	12	11	0	**0.95**	58	57	43	0.29	0.14	0.20	0.64	-0.14
$ec_{11} : [h_1, w_6 \rightarrow w_7, v_1]$	S	0.82	5.79	0.77	0.25	**0.88**	36	30	28	2	**0.89**	109	164	102	**0.96**	N/A	N/A	N/A	N/A
$ec_{12} : [h_1, w_6 \rightarrow w_8, v_1]$	S	**1.00**	0.52	**0.92**	**0.80**	**0.97**	38	30	22	6	0.94	51	53	43	0.99	N/A	N/A	N/A	N/A
$ec_{13} : [h_1, w_8 \rightarrow w_7, v_1]$	S	**1.00**	0.32	**0.92**	0.53	**0.99**	30	33	26	1	0.98	53	89	51	1.00	N/A	N/A	N/A	N/A
$ec_{14} : [h_1, w_9 \rightarrow w_{10}, v_1]$	L	0.24	4.85	0.56	0.44	0.77	22	21	18	3	0.69	**237**	**226**	**94**	**0.86**	N/A	N/A	N/A	N/A

ES: Expected severity of change (Sec. III-B): **S**: small change; **SM**: small medium change; **M**: medium change; **L**: large change; **VL**: very large change.
SaC workload descriptions: srad: random matrix generator; pfilter: particle filtering; hotspot: heat transfer differential equations; k-means: clustering; nw: optimal matching; nbody: simulation of dynamic systems; cg: conjugate gradient; gc: garbage collector. Hardware descriptions (ID: Type/CPUs/Clock (GHz)/RAM (GiB)/Disk):
h1: NUC/4/1.30/15/SSD; **h2**: NUC/2/2.13/7/SCSI; **h3**:Station/2/2.8/3/SCSI; **h4**: Amazon/1/2.4/1/SSD; **h5**: Amazon/1/2.4/0.5/SSD; **h6**: Azure/1/2.4/3/SCSI
Metrics: **M1**: Pearson correlation; **M2**: Kullback-Leibler (KL) divergence; **M3**: Spearman correlation; **M4/M5**: Perc. of top/bottom conf.; **M6/M7**: Number of influential options; **M8/M9**: Number of options agree/disagree; **M10**: Correlation btw importance of options; **M11/M12**: Number of interactions; **M13**: Number of interactions agree on effects;

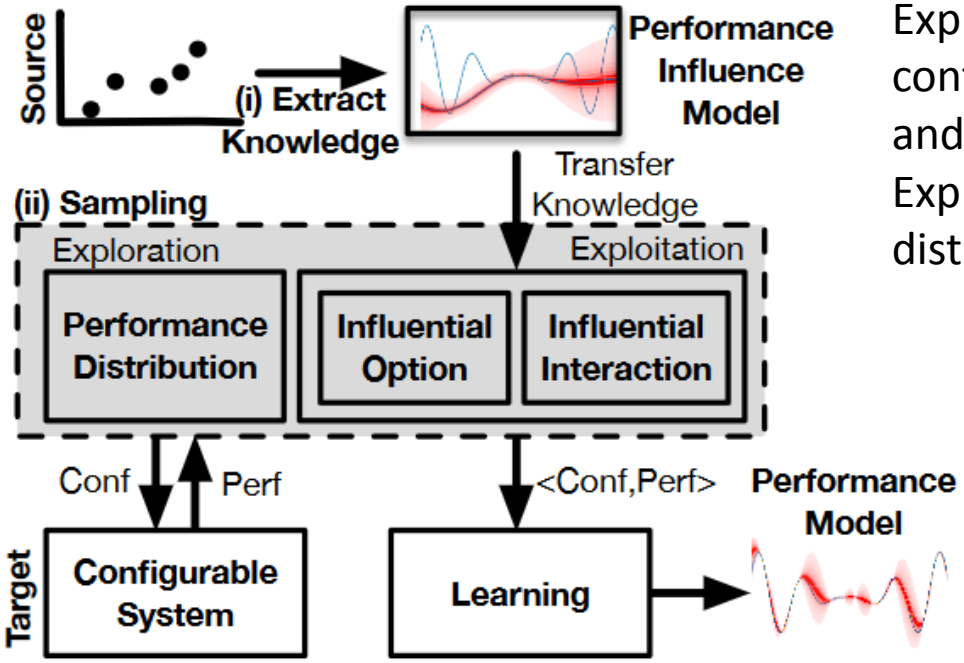
Insight 1. **Performance distributions can be transferred:** Potential for learning a non-linear transfer function.

Insight 2. **Configuration ranks can be transferred:** Good configurations stay good for changing hardware.

Insight 3. **Influential options and interactions can be transferred:** Relevant options in one environment stay relevant in other environments.

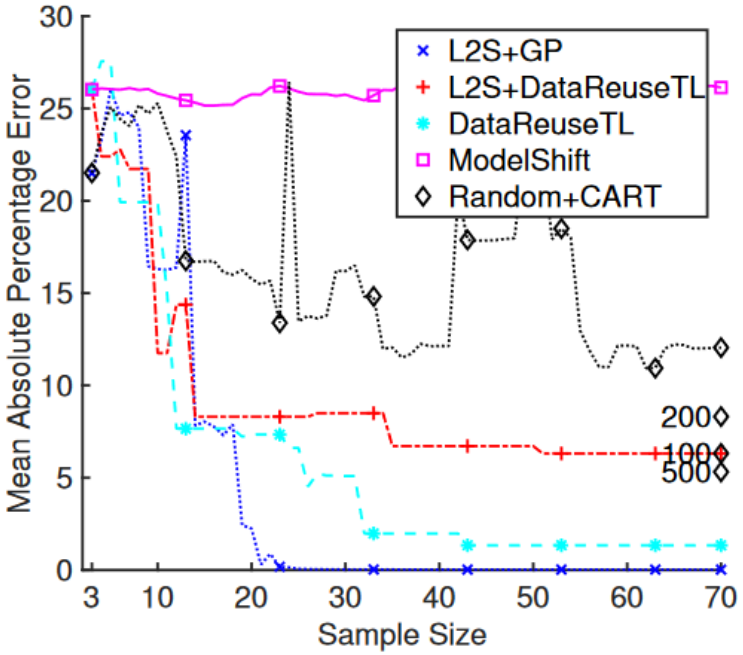
Learn To Sample

Learn model efficiently in changed environment: [54] Jamshidi et al. FSE'18: Active transfer learning



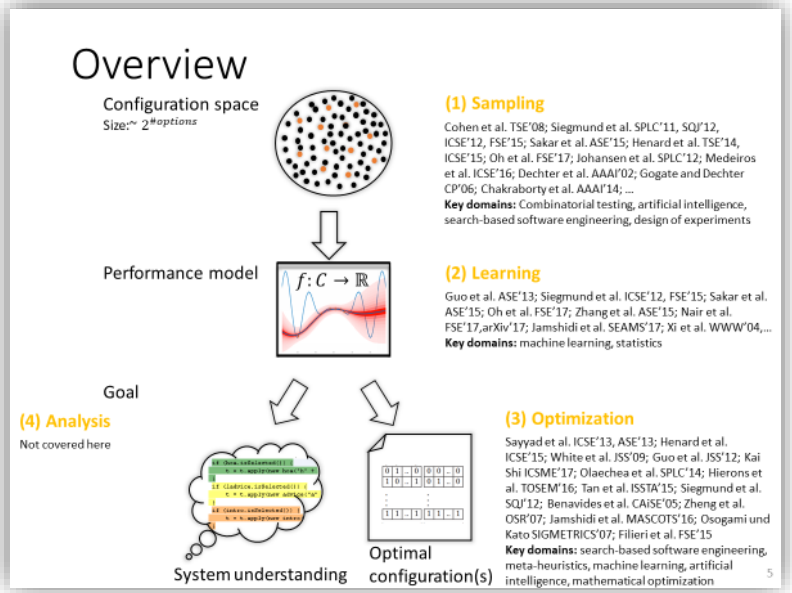
Exploitation: Generate priority list of configurations based on influential options and interactions

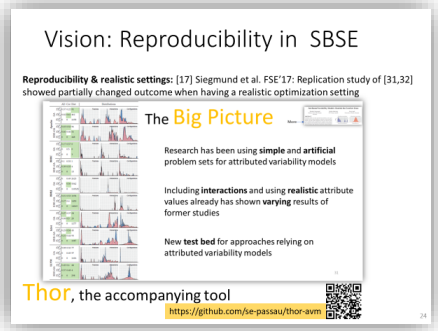
Exploration: Sampling based on performance distribution of configurations

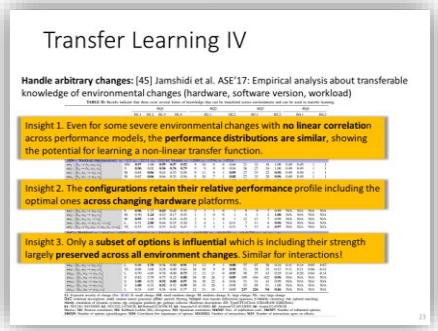


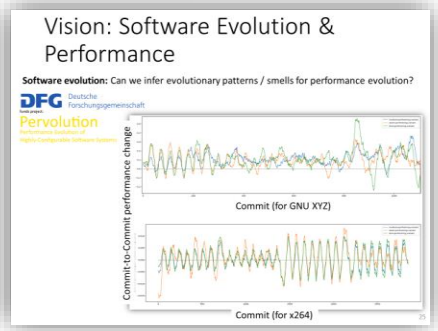
Use of synthetic models for internal validity and real-world subject systems for external validity

Summary









- Other domains, similar problems (i.e., not covered):
  - Active learning (e.g., [27,28] Zuluaga et al. JML'16,ICML'13)
  - Parameter optimization / tuning, algorithm selection (e.g., [29] Hutter et al. LION'05; [30] Wu et al. GECCO'15)
  - Self-adaptive systems (e.g., [46] Filieri et al. FSE'15)
  - Systems optimization (e.g., [47] Osogami et al. SIGMETRICS'07, [48] Zheng et al. OSR'07, [49] Xi et al. WWW'04)

DFG Deutsche Forschungsgemeinschaft funds project: Green Configuration
Understanding the Influence of Software Configurations on Energy Consumption

DFG Deutsche Forschungsgemeinschaft funds project: Pervolution
Performance Evolution of Highly-Configurable Software Systems