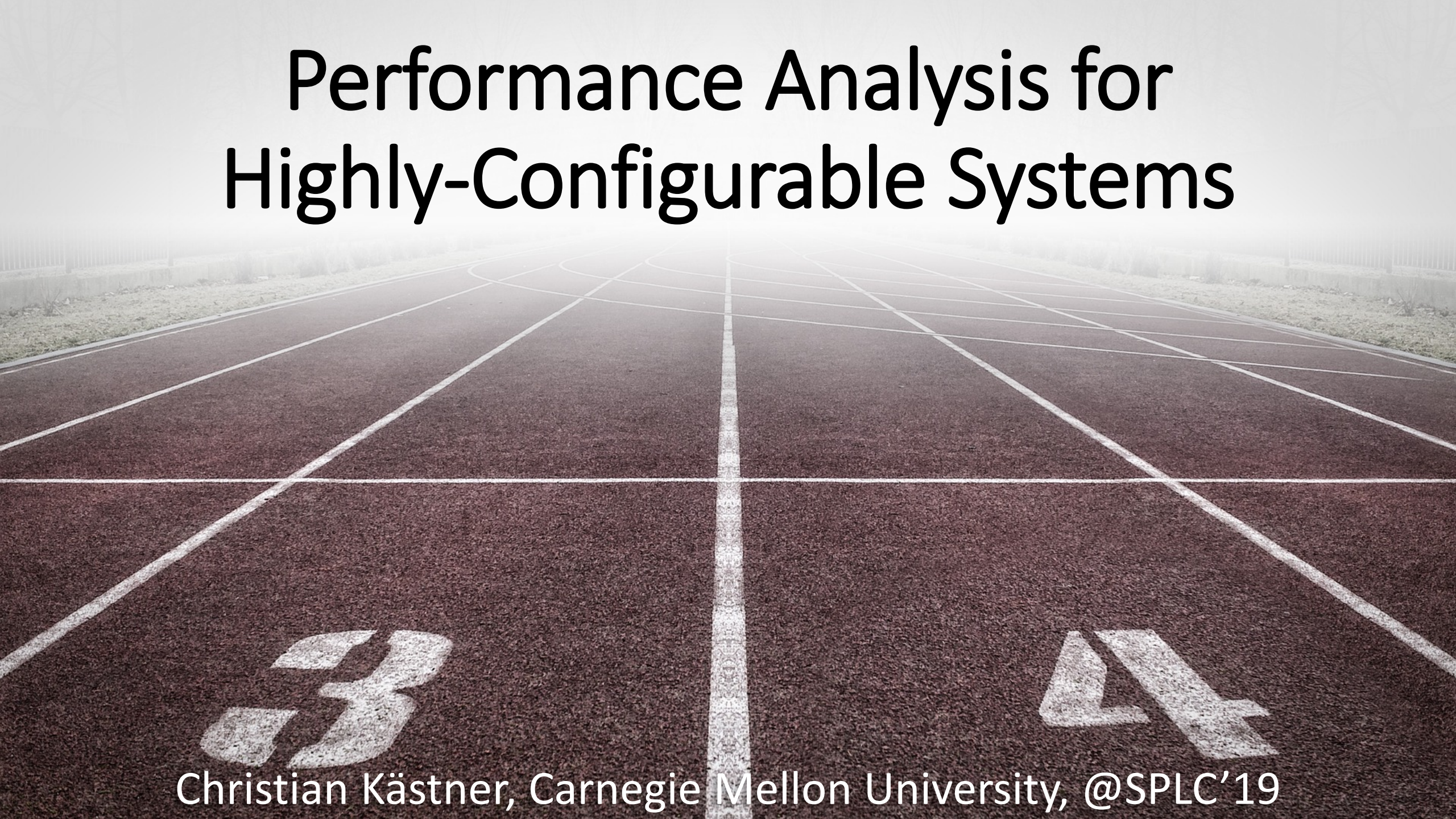
Performance Analysis for Highly-Configurable Systems



Christian Kästner, Carnegie Mellon University, @SPLC'19


96
AMTRAK
AMTRAK
96

Collaborators

Norbert Siegmund
University of Weimar

Sven Apel
Saarland University

Miguel Velez
Carnegie Mellon Univ.

Pooyan Jamshidi
Univ. South Carolina

David Garlan
Carnegie Mellon Univ.

Sergiy Kolesnikov
University of Passau

Case Study

“**SocialSensor** quickly surfaces trusted and relevant material from social media - with context.”

Internet

Crawled items

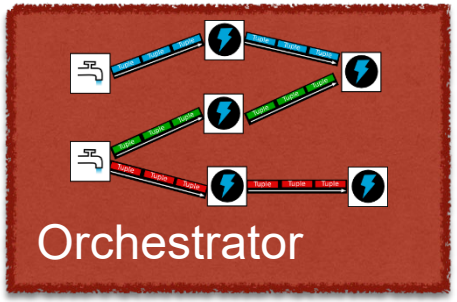
Tweets: [5k-20k/min]

Store

cassandra

Fetch

Every 10 min:
[100k tweets]



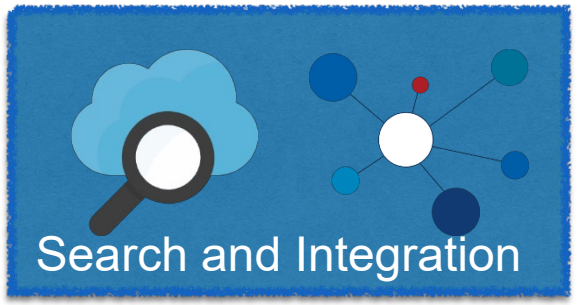
Push

Store

Solr

Tweets: [10M]

Fetch



Changing Requirements

Internet

Crawled items

10X

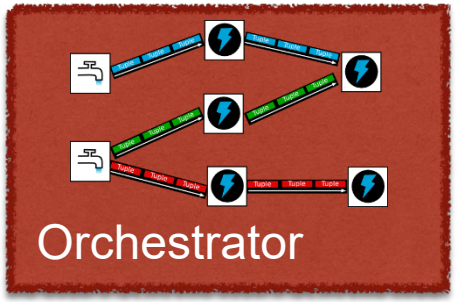
Tweets: [5k-20k/min]

Store

Fetch

Real time

Every 10 min:
[100k tweets]



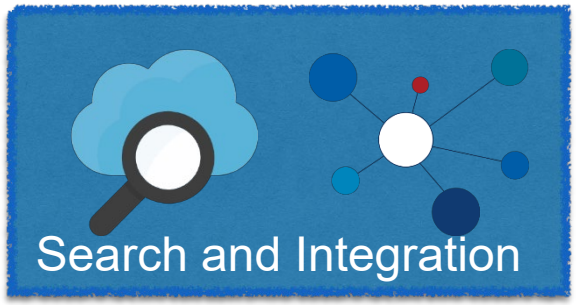
Push

Store

Tweets: [10M]

Fetch

100X



Opportunity

Data processing engines highly configurable

cassandra

> 100 options

STORM

> 100 options

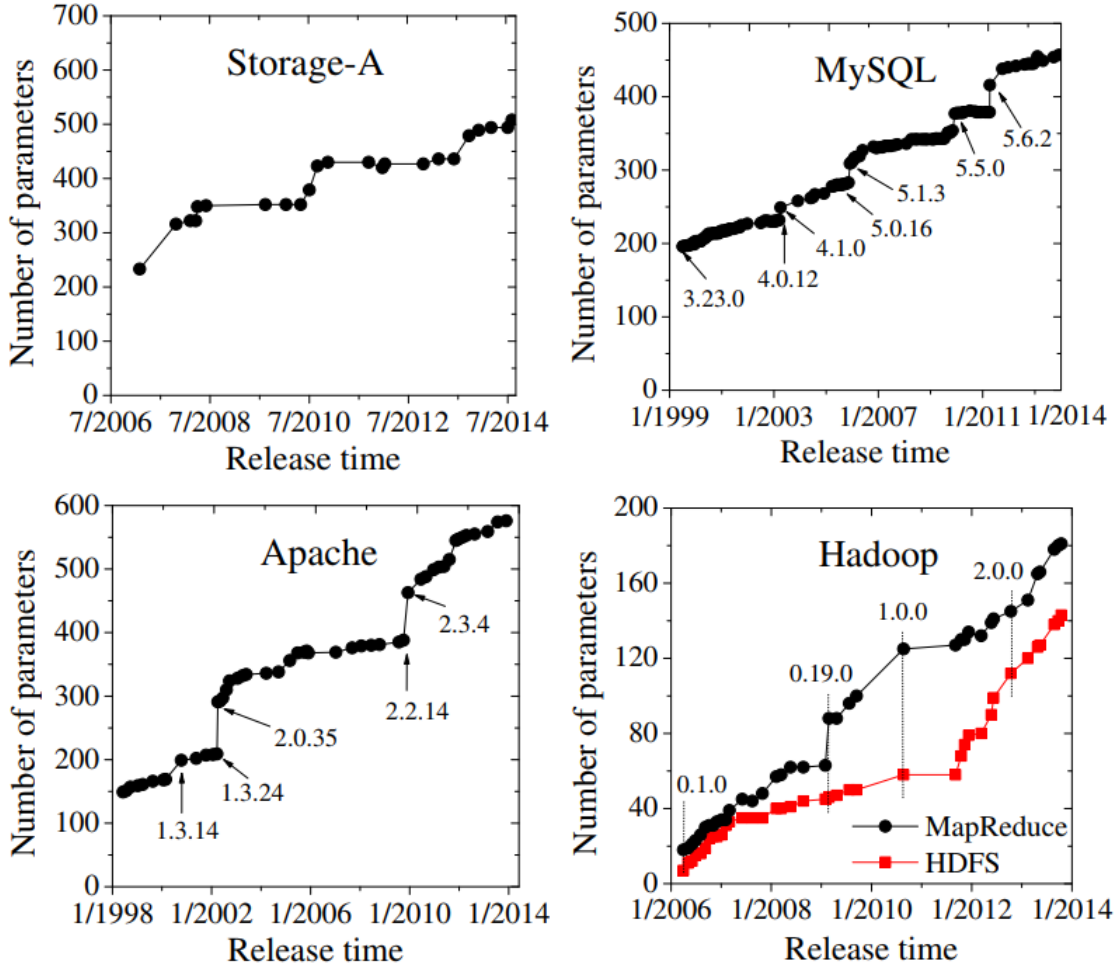
Solr

> 100 options

```
102
103 drpc.port: 3772
104 drpc.worker.threads: 64
105 drpc.max_buffer_size: 1048576
106 drpc.queue.size: 128
107 drpc.invocations.port: 3773
108 drpc.invocations.threads: 64
109 drpc.request.timeout.secs: 600
110 drpc.childopts: "-Xmx768m"
111 drpc.http.port: 3774
112 drpc.https.port: -1
113 drpc.https.keystore.password: ""
114 drpc.https.keystore.type: "JKS"
115 drpc.http.creds.plugin: org.apache.storm.security.auth.DefaultHttpCredentialsPlugi
116 drpc.authorizer.acl.filename: "drpc-auth-acl.yaml"
117 drpc.authorizer.acl.strict: false
118
119 transactional.zookeeper.root: "/transactional"
120 transactional.zookeeper.servers: null
121 transactional.zookeeper.port: null
122
123 ## blobstore configs
124 supervisor.blobstore.class: "org.apache.storm.blobstore.NimbusBlobStore"
125 supervisor.blobstore.download.thread.count: 5
126 supervisor.blobstore.download.max_retries: 3
127 supervisor.localizer.cache.target.size.mb: 10240
128 supervisor.localizer.cleanup.interval.ms: 600000
129
```


320 optional, independent options
more combinations than estimated
atoms in the universe

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



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

80% options ignored

Changing Requirements

Internet

Crawled items

10X

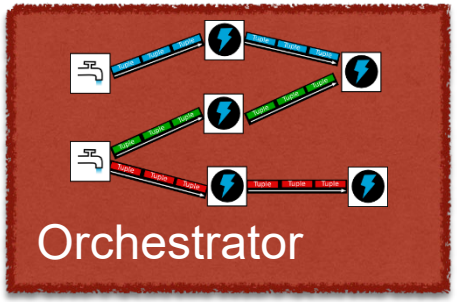
Tweets: [5k-20k/min]

Store

Fetch

Real time

Every 10 min:
[100k tweets]



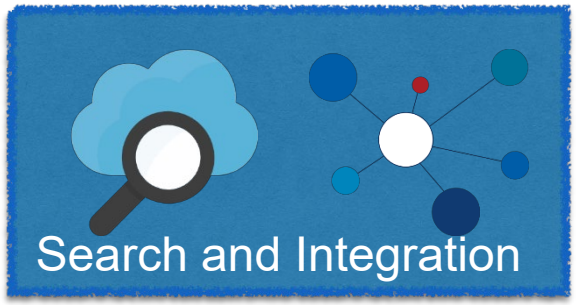
Push

Store

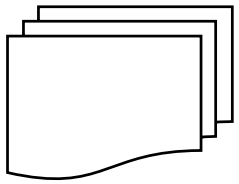
Tweets: [10M]

Fetch

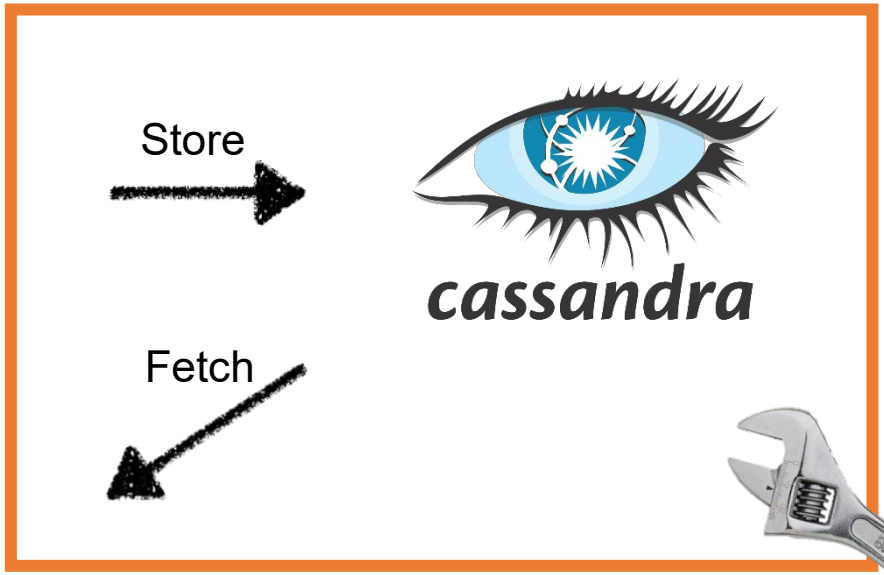
100X



Experiment

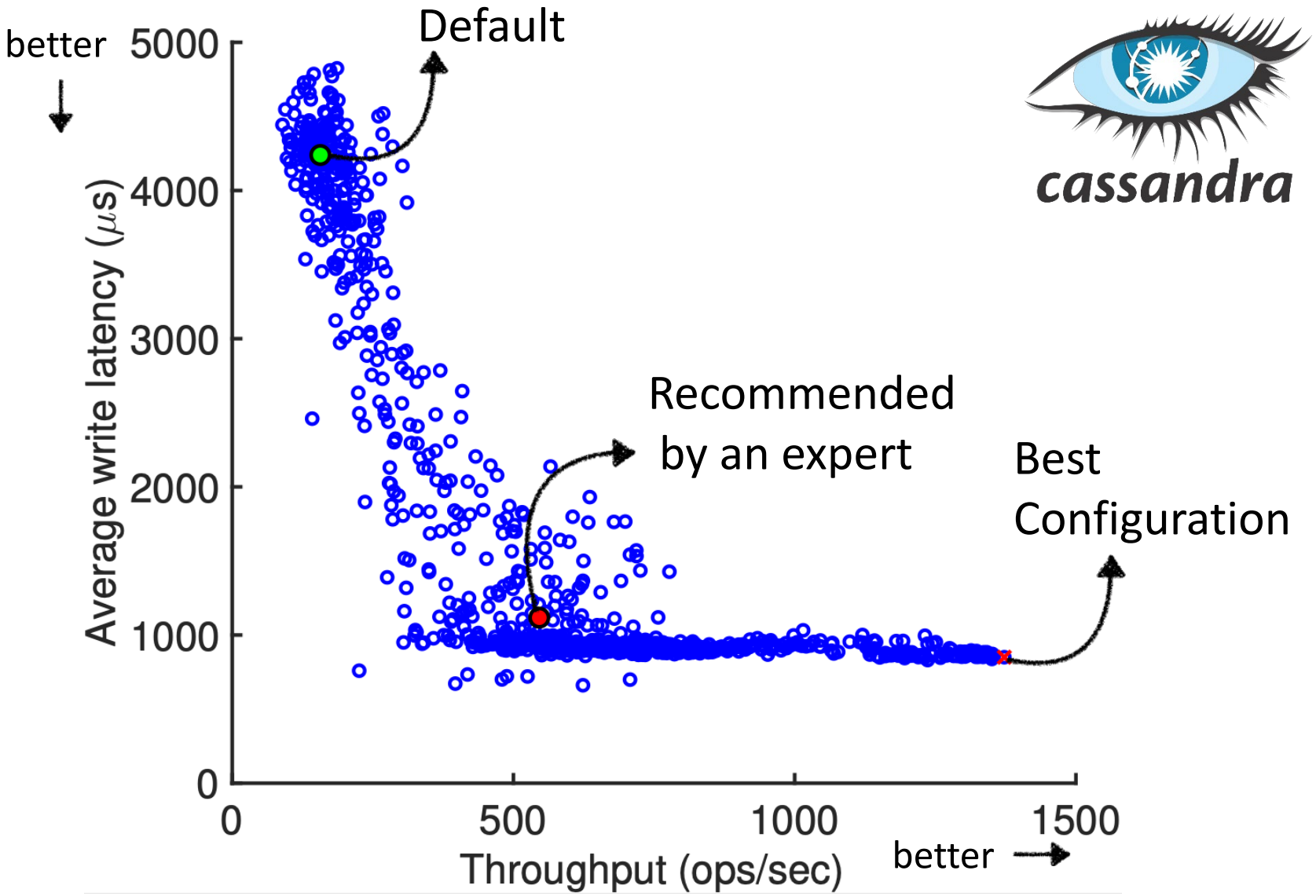


Representative workload

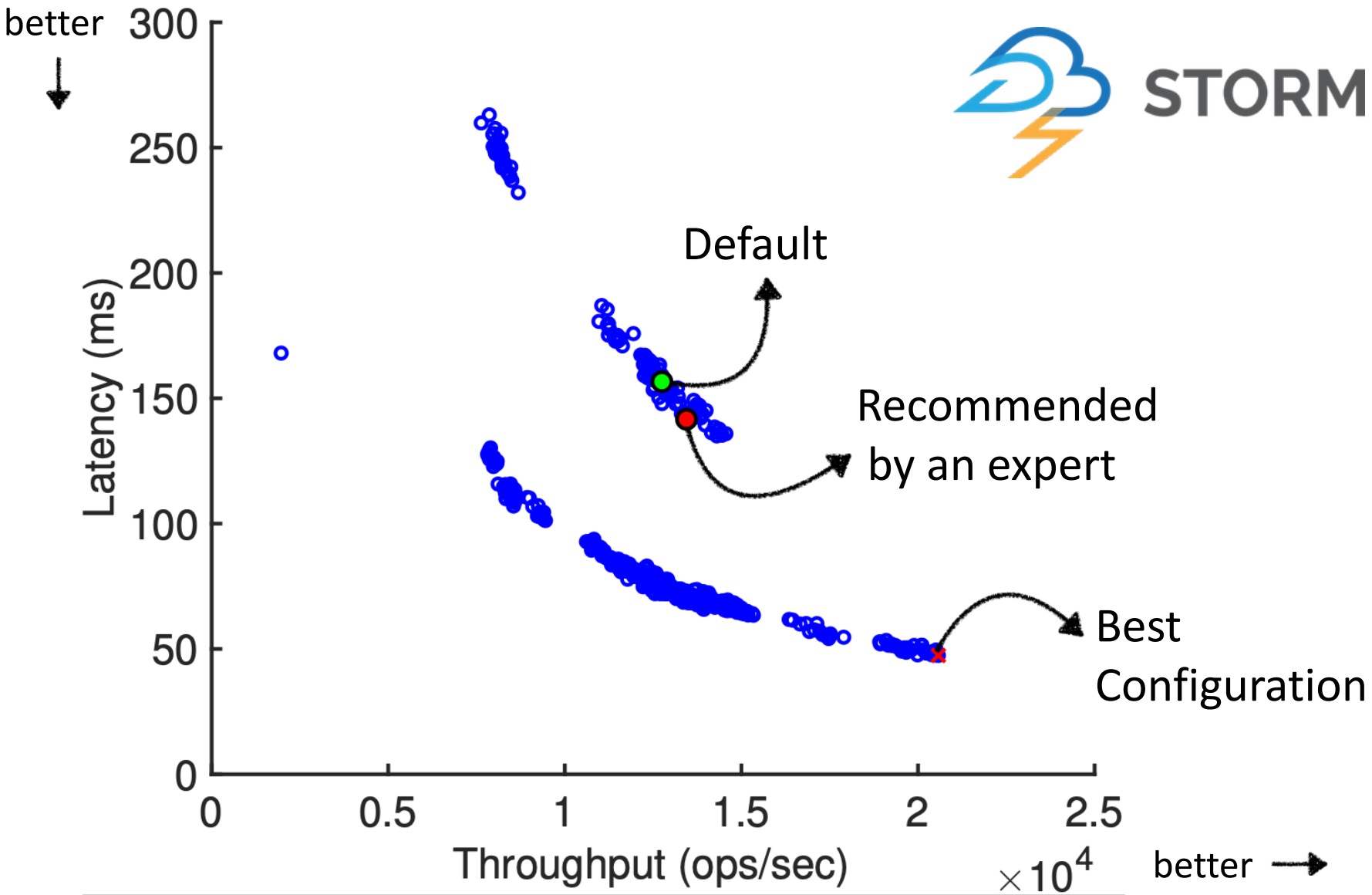


Performance measure of interest

Change configuration


better
5000
4000
3000
2000
1000
0
Average write latency (μs)
Default
Recommended
by an expert
Best
Configuration
0
500
1000
1500
Throughput (ops/sec)
better

cassandra


better
300
250
200
150
100
50
0
Latency (ms)
0
0.5
1
1.5
2
2.5
Throughput (ops/sec)
$\times 10^4$
better
STORM
Default
Recommended
by an expert
Best
Configuration

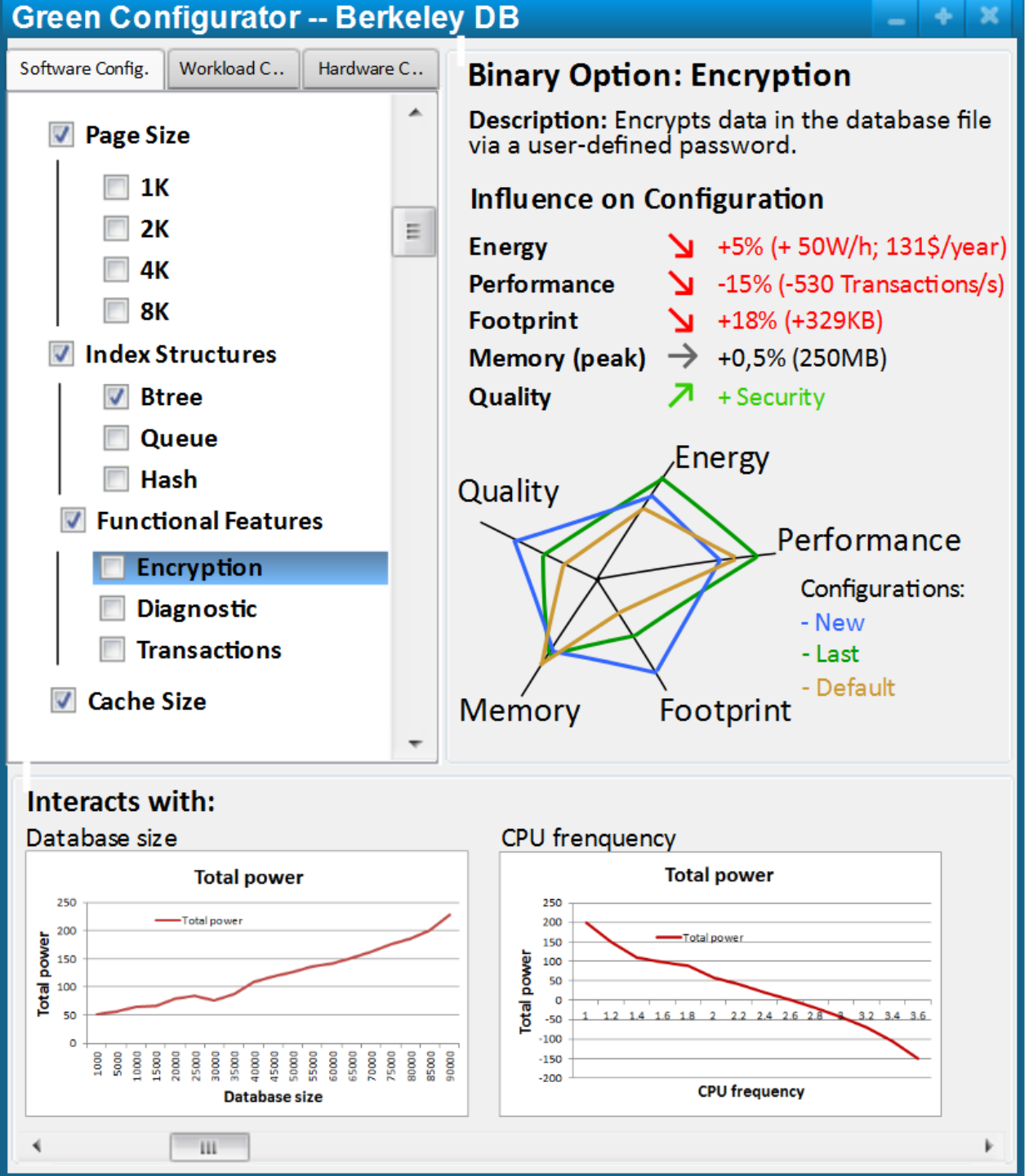

100X more user
cloud resources reduced **20%**
outperform expert recommendation

Why Options?

Design for performance vs configuration

No one size fits all
Workload, hardware differs
Different users make different tradeoffs

Options are deferred design decisions



Example

Index

Encryption

Compression

Logging

Write performance

Query performance

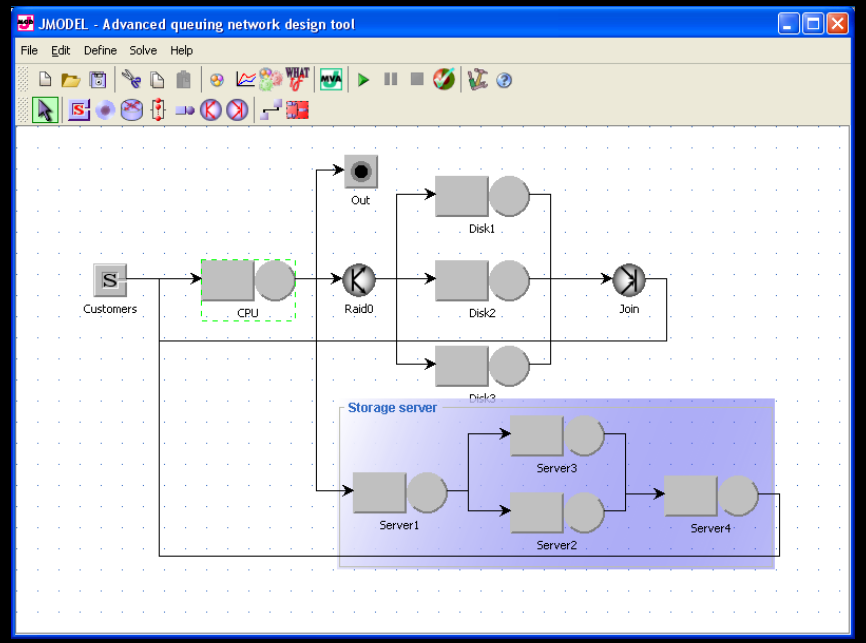
Scalability

Security

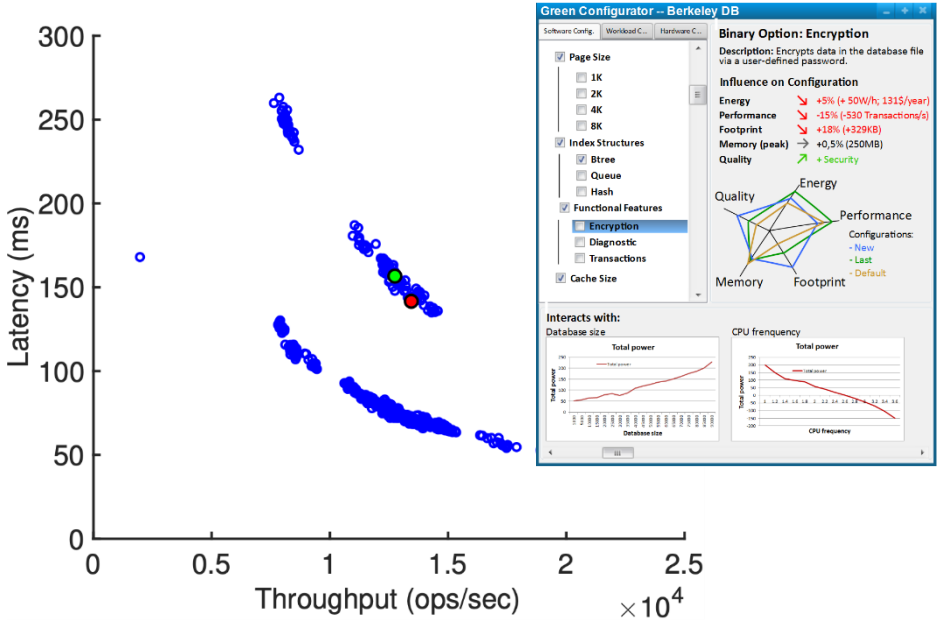
...

Performance Analysis: Goals

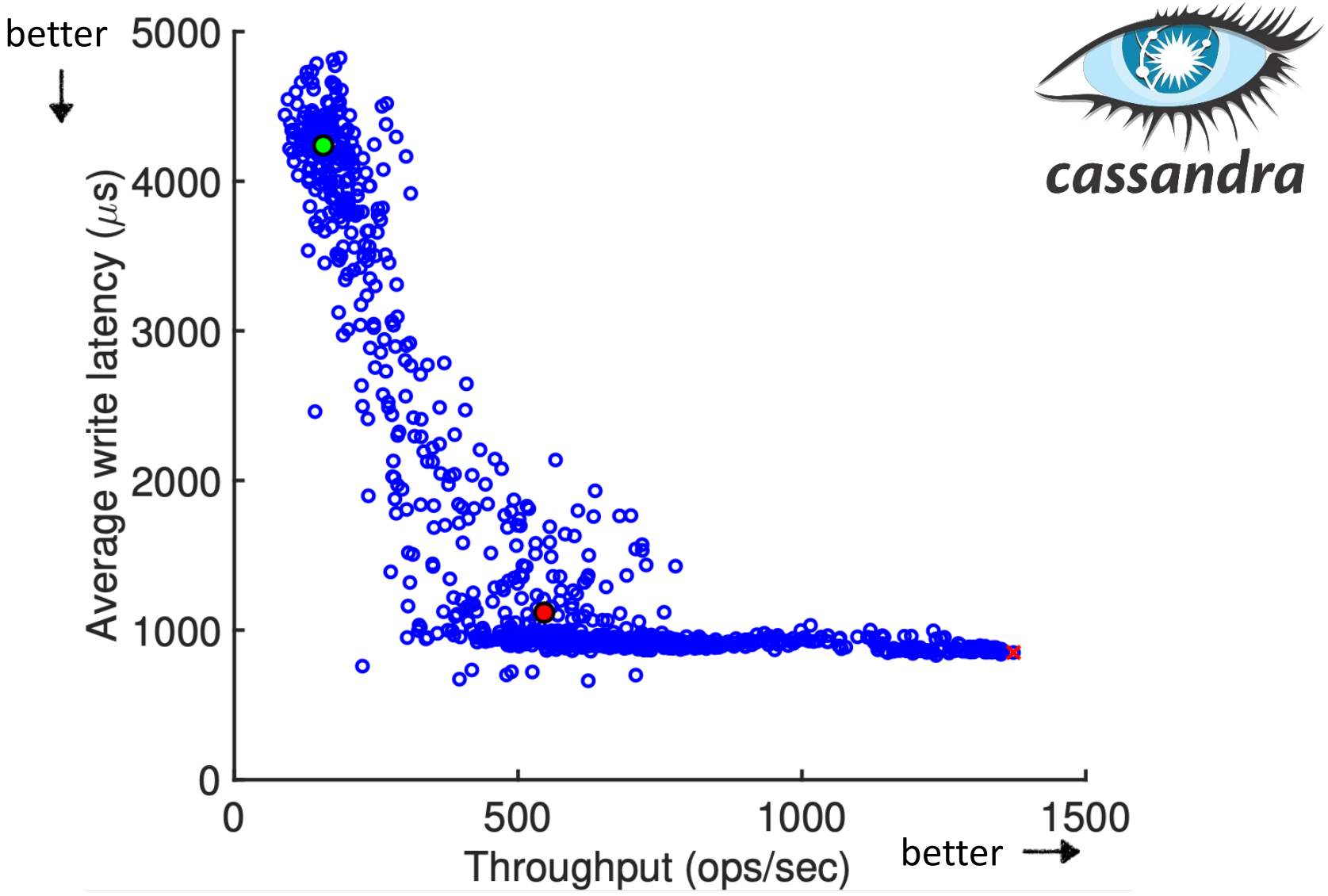
Performance Modeling for Design



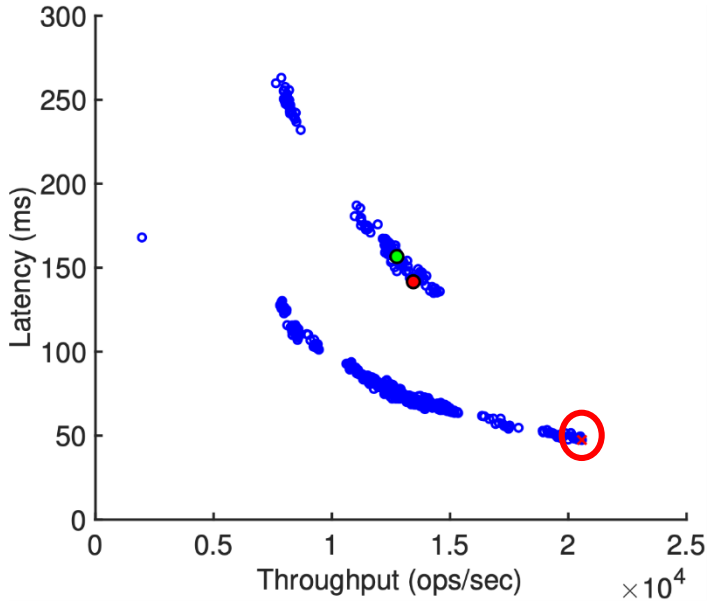
Optimization & Model Inference for Existing Code



Optimization


better
5000
4000
3000
2000
1000
0
Average write latency (μs)
0
500
1000
1500
Throughput (ops/sec)
better
cassandra

Optimization



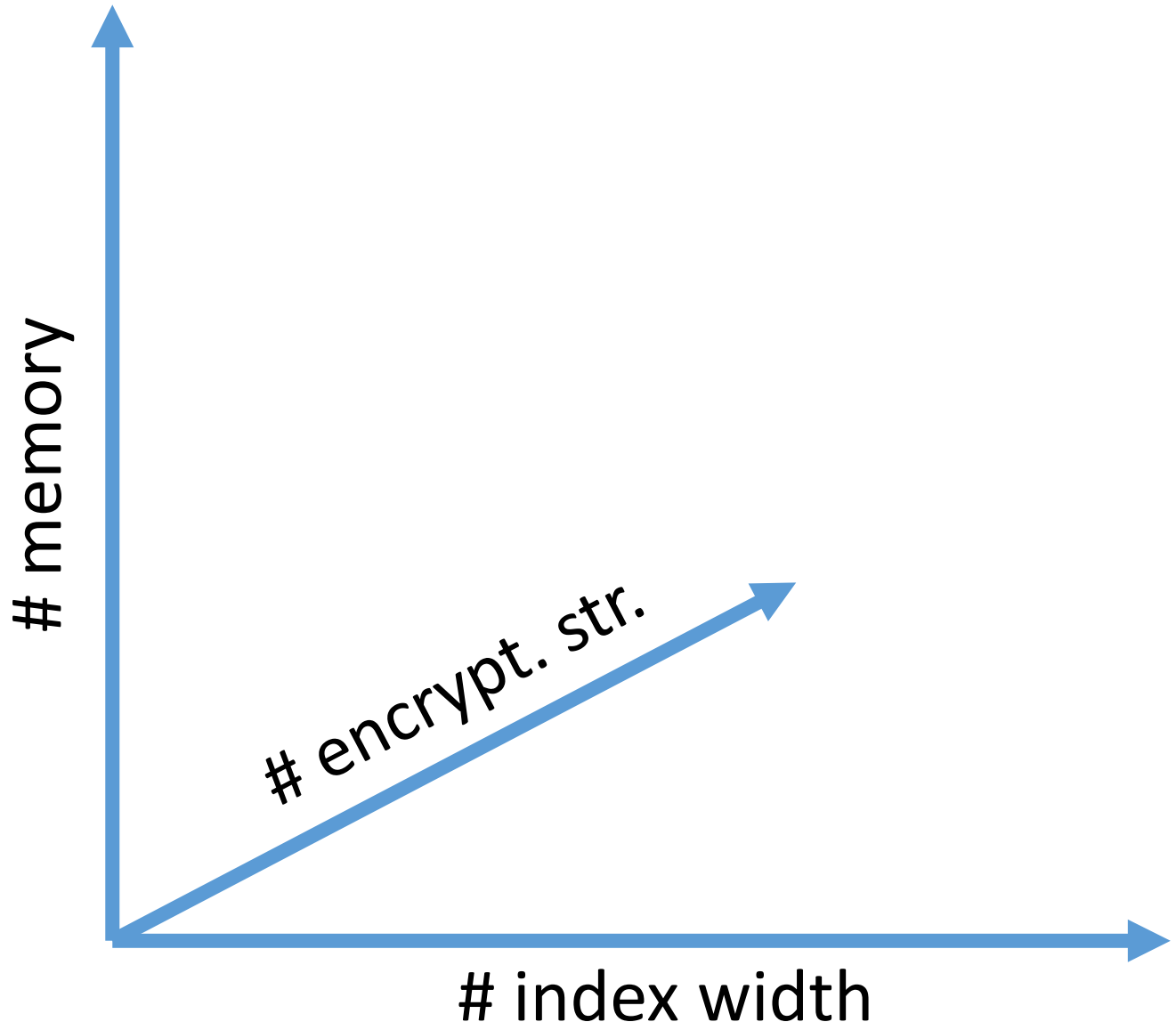
Goal: Find the fastest configuration or optimize a fitness function

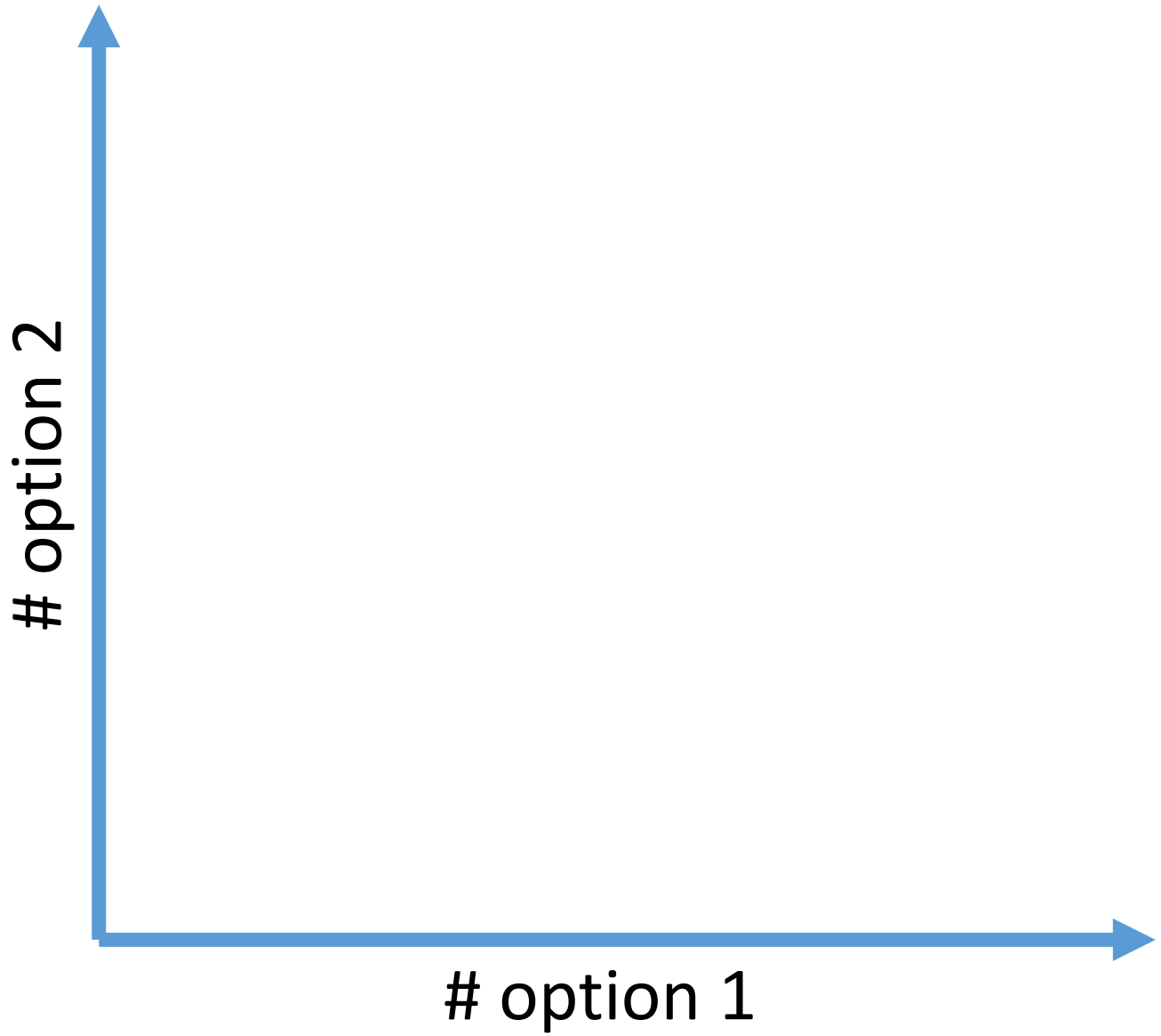
Search problem, implementation given

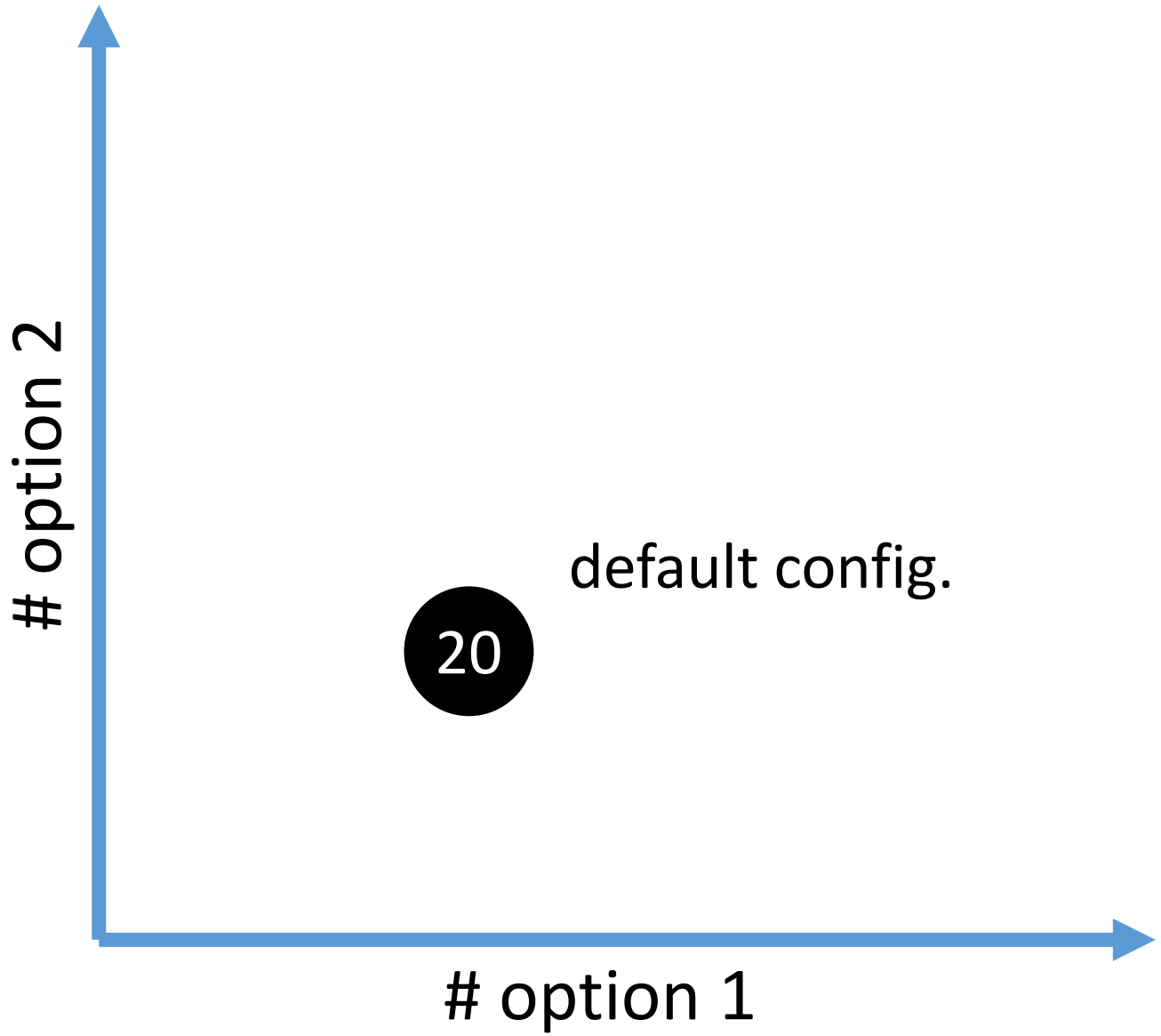
Explored in many communities (databases, AI, systems, SE, …)

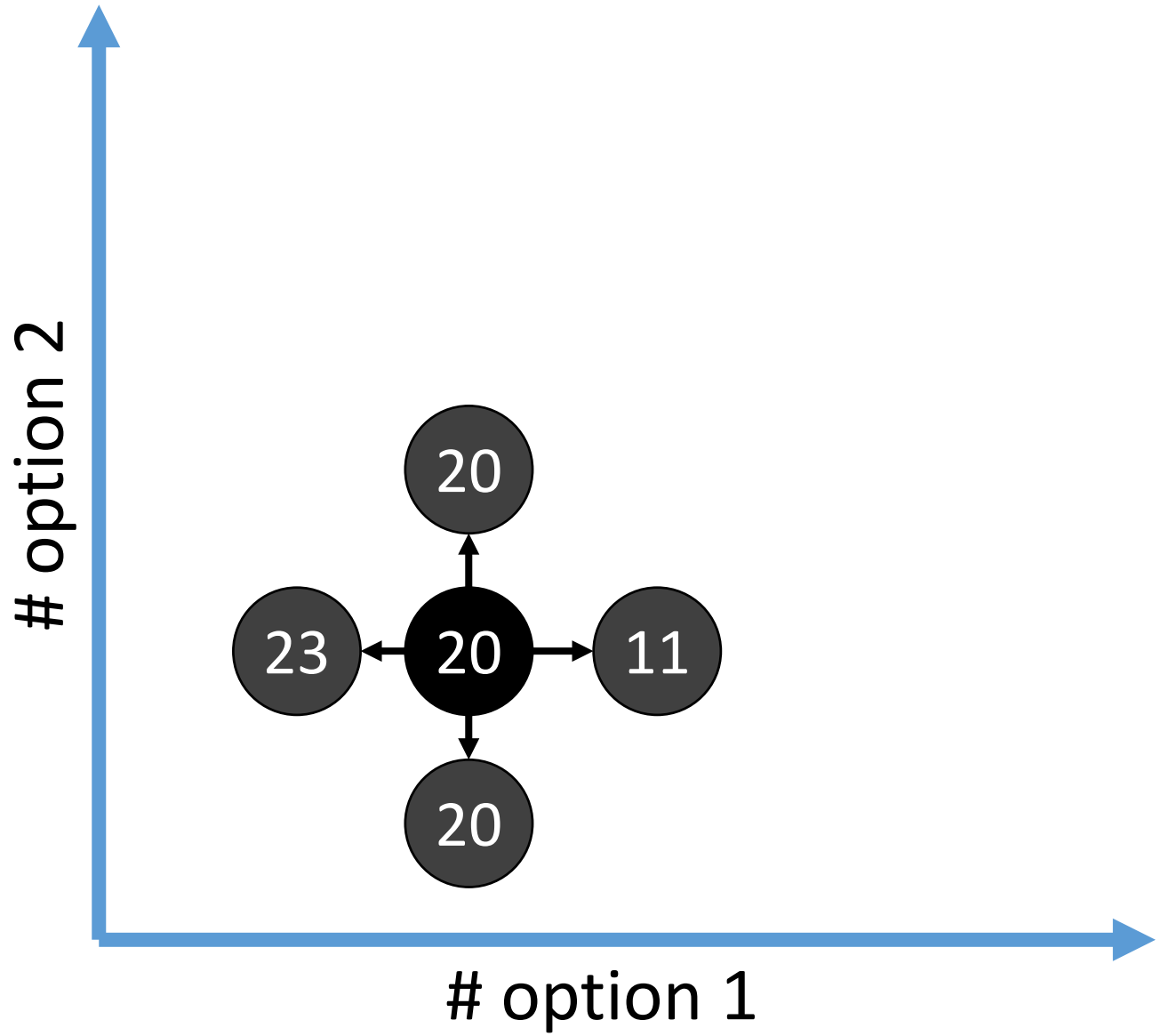
Auto-tuning, hyperparameter optimization, algorithm selection

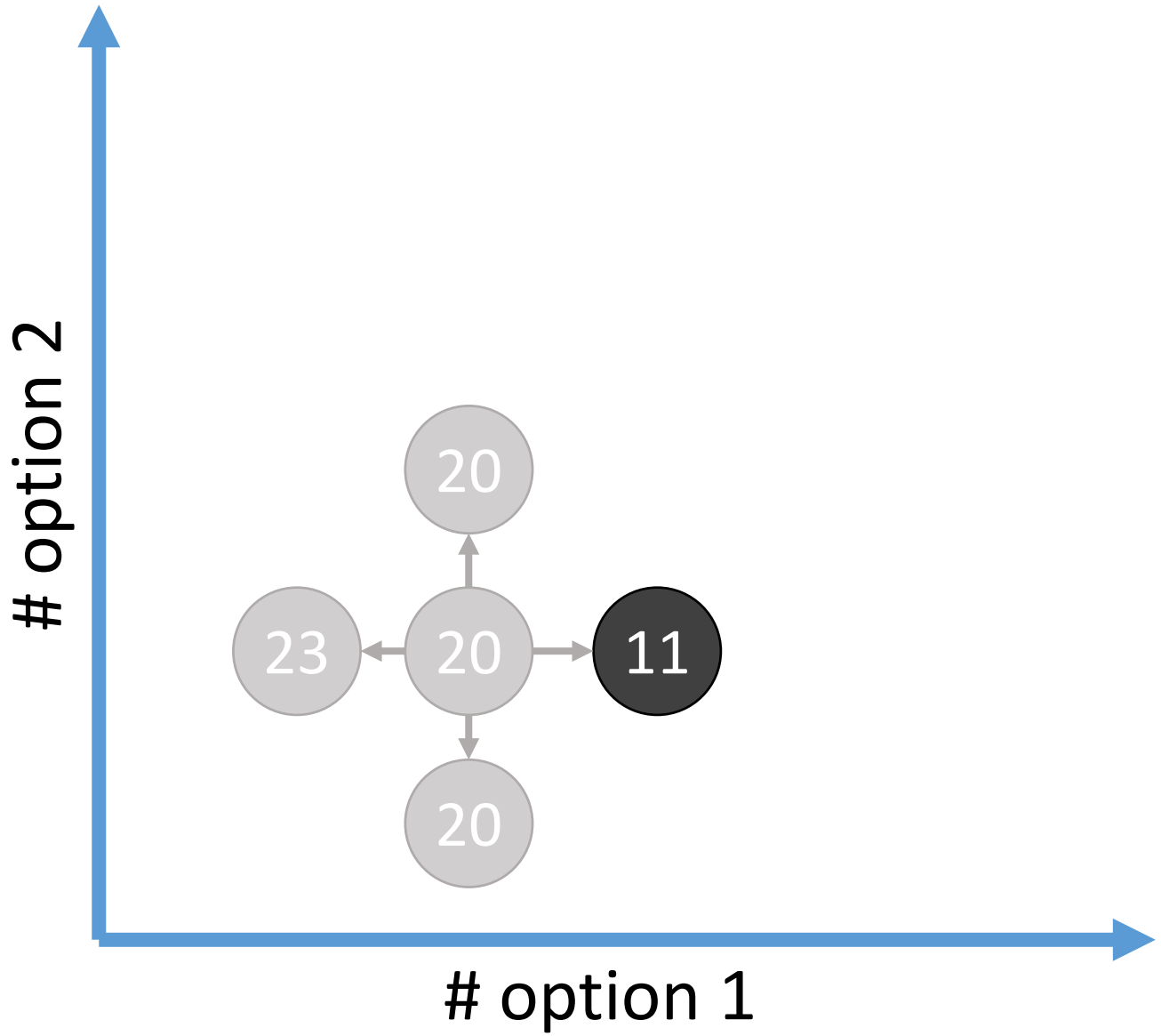
Evaluated by performance improvement or distance to optimum

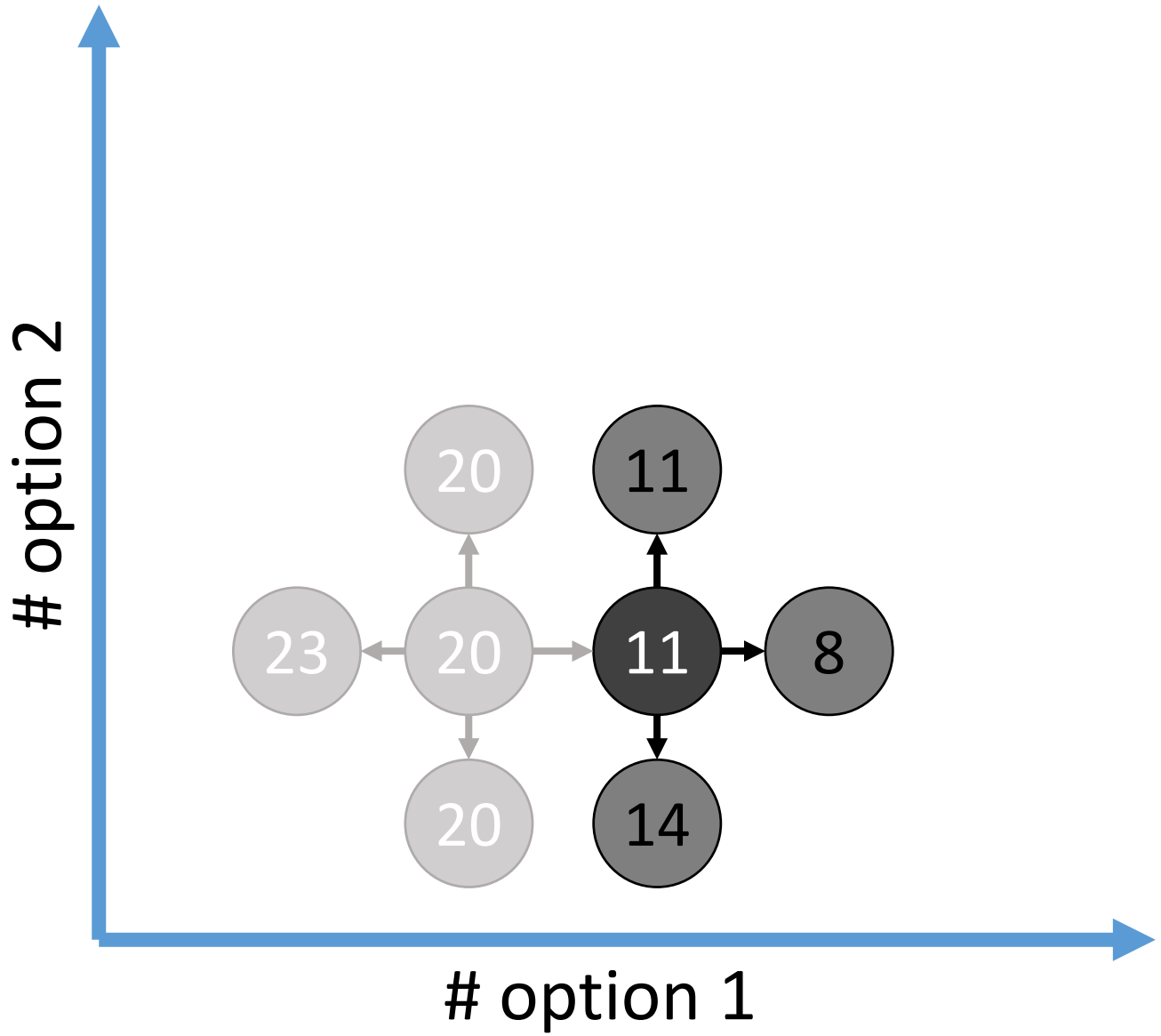

memory
encrypt. str.
index width


option 2
option 1

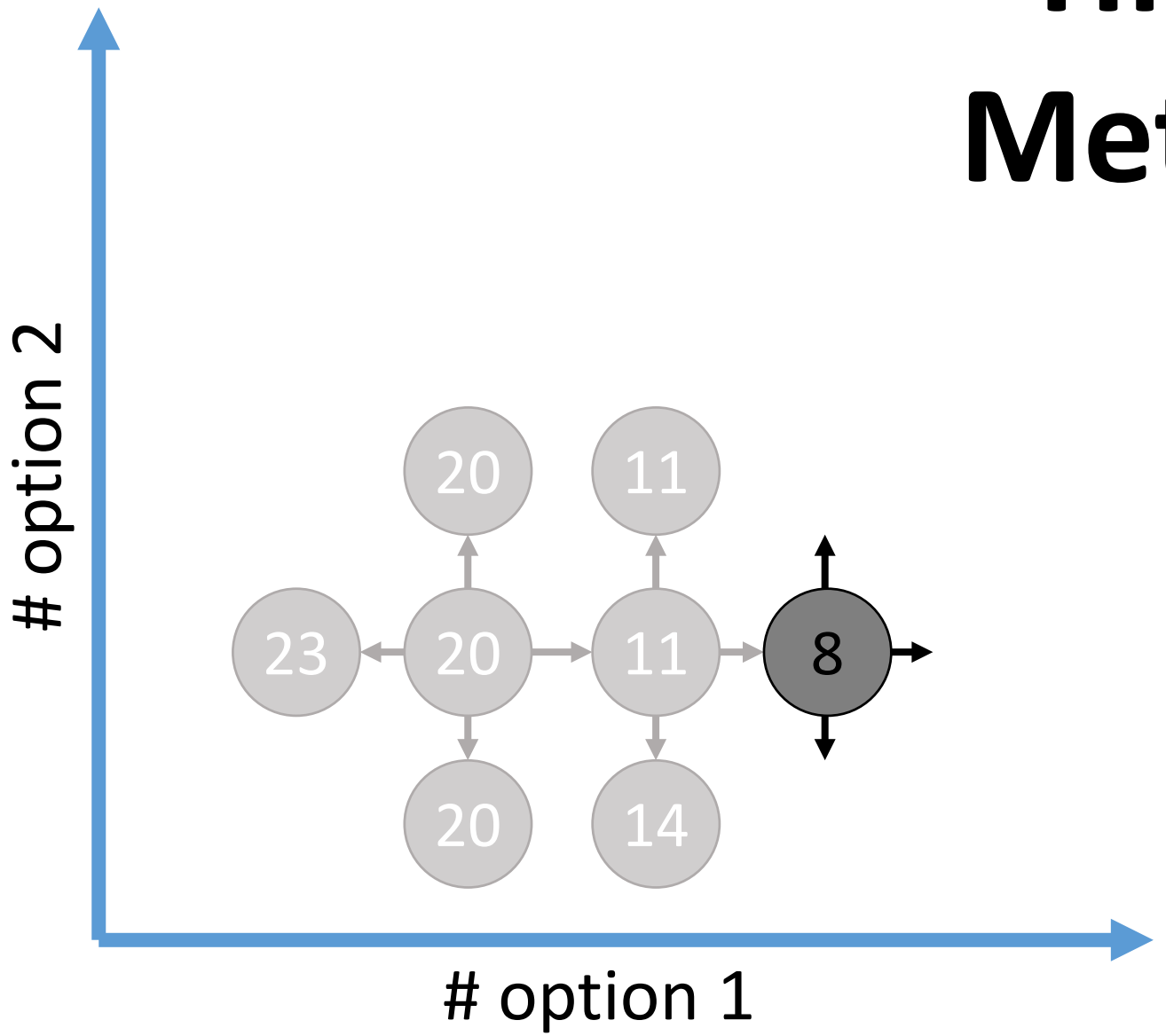

option 2
default config.
20
option 1

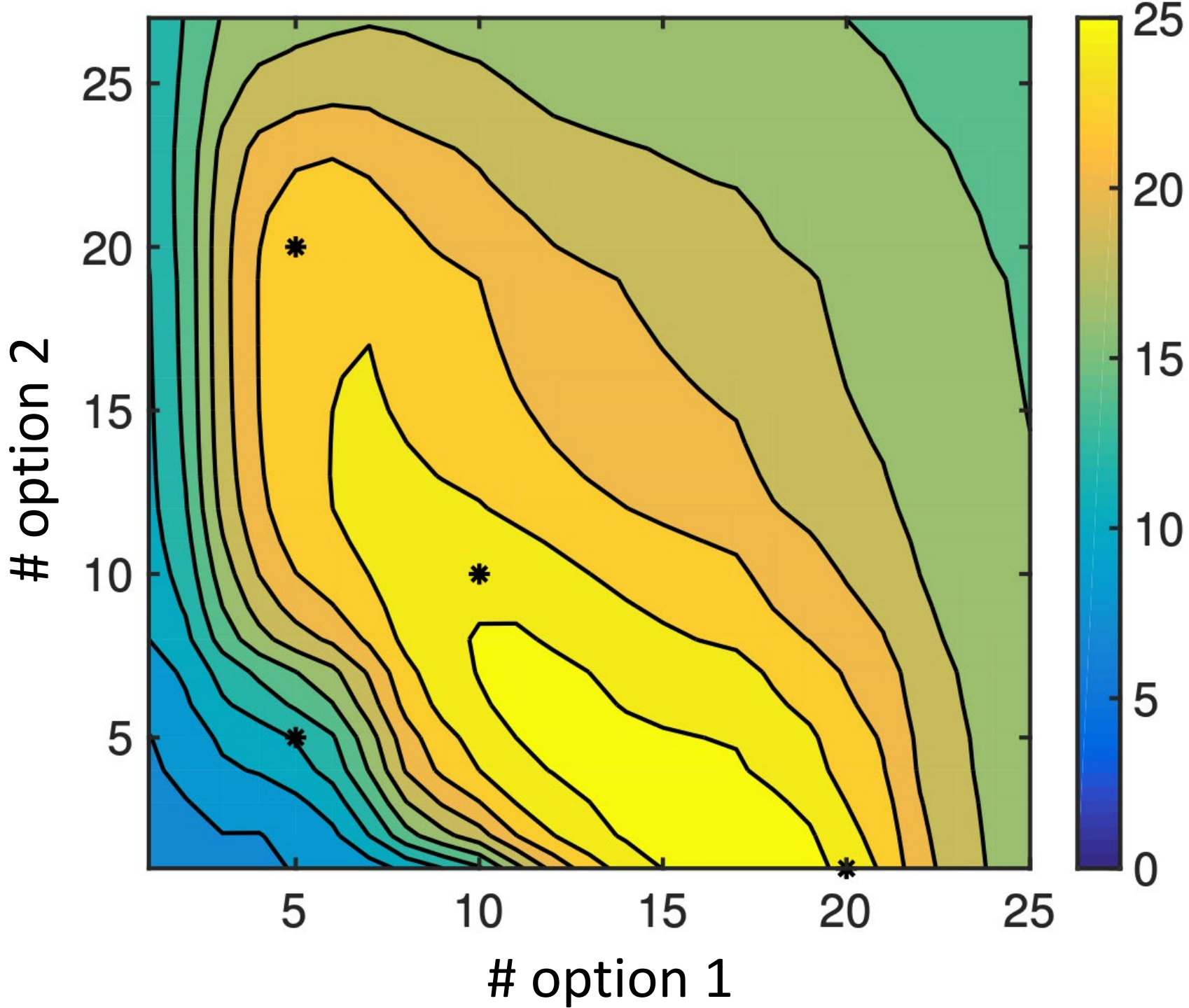

option 2
20
23
20
11
20
option 1


option 2
20
23
20
11
20
option 1


option 2
20
11
23
20
11
8
20
14
option 1

Hill Climbing Metaheuristic




25
20
15
10
5
option 2
5
10
15
20
25
option 1
25
20
15
10
5
0

Optimizing Selection of Competing Features via Feedback-Directed Evolutionary Algorithms

Tian Huat Tan† Yinxing Xue* Manman Chen* Jun Sun† Yang Liu‡ Jin Song Dong*
†Singapore University of Technology and Design, Singapore
*National University of Singapore, Singapore
‡Nanyang Technological University, Singapore

Faster discovery of faster system configurations with spectral learning

Vivek Nair[1] · Tim Menzies[1] ·
Norbert Siegmund[2] · Sven Apel[3]

SIP: Optimal Product Selection from Feature Models Using Many-Objective Evolutionary Optimization

ROBERT M. HIERONS, MIQING LI, and XIAOHUI LIU, Brunel University London, UK
SERGIO SEGURA, University of Seville, Spain
WEI ZHENG, Northwestern Polytechnical University, China

Combining Evolutionary Algorithms with Constraint Solving for Configuration Optimization

Sequential Model-Based Optimization for General Algorithm Configuration

Frank Hutter, Holger H. Hoos, and Kevin Leyton-Brown

Comparison of Exact and Approximate Multi-Objective Optimization for Software Product Lines

Rafael Olaechea, Derek Rayside, Jianmei Guo, Krzysztof Czarnecki
University of Waterloo
Waterloo, Ontario
{rolaechea, gjm,kczarnec}@gsd.uwaterloo.ca, {drayside}@uwaterloo.ca

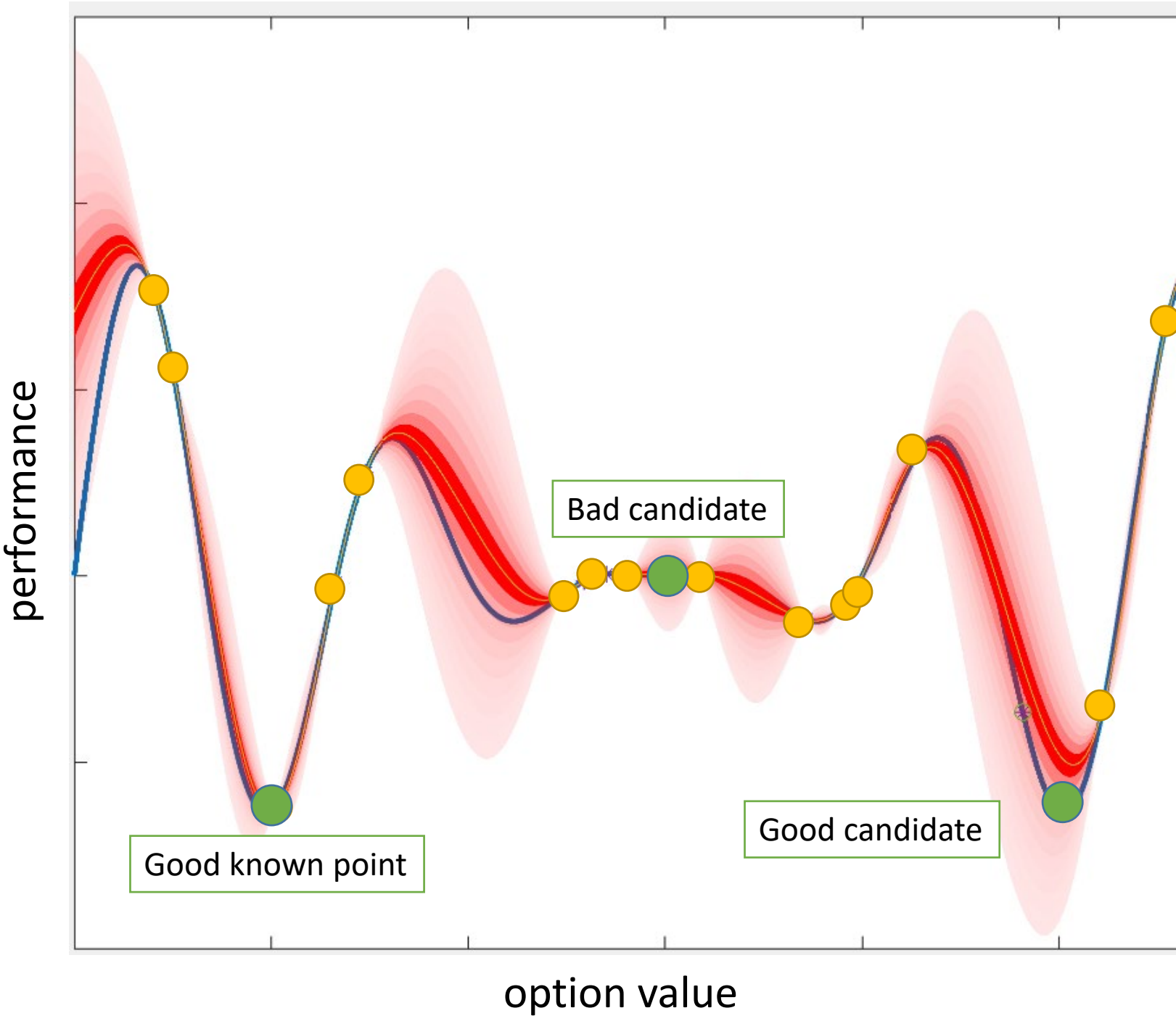
Random Search for Hyper-Parameter Optimization

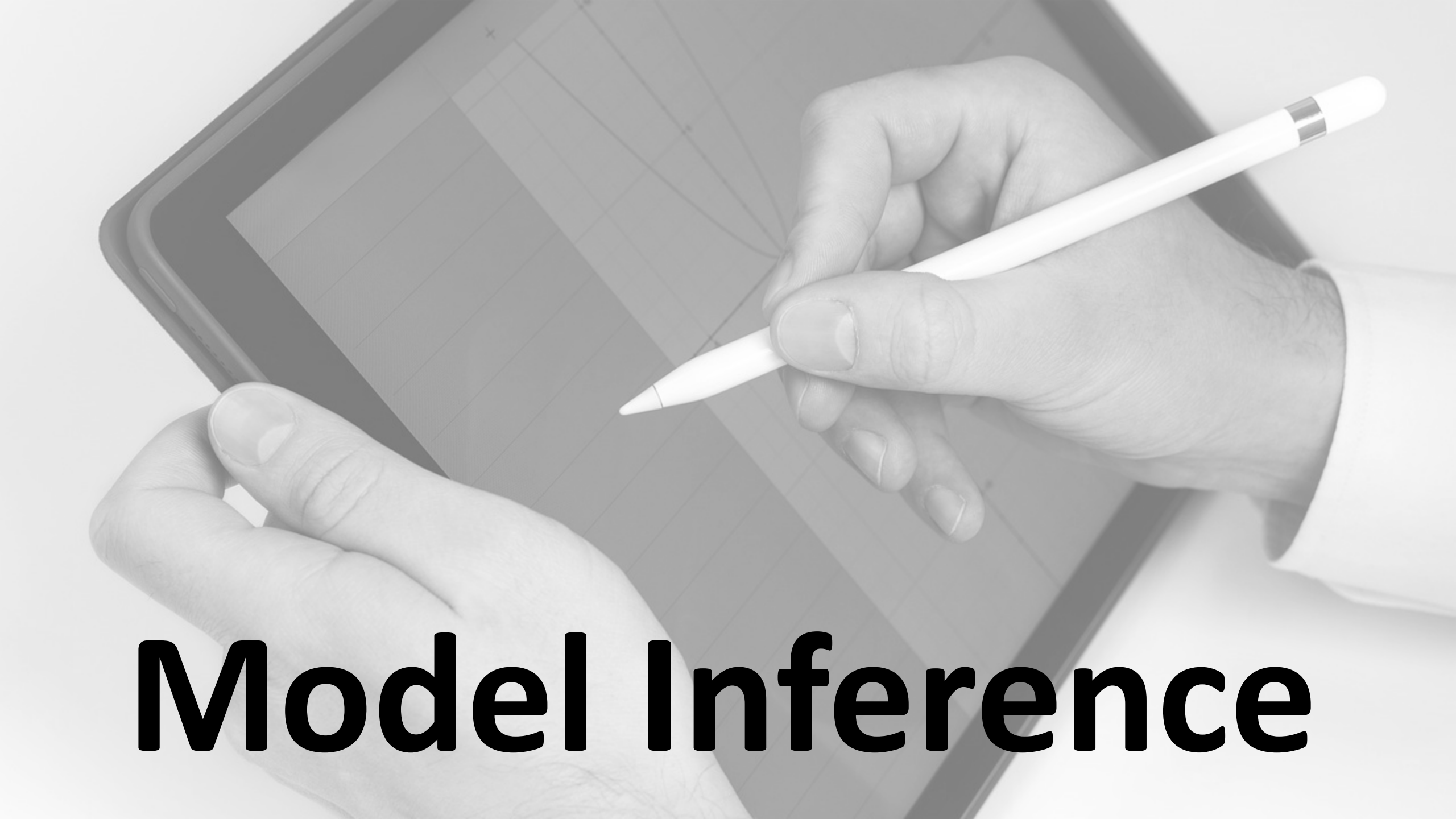
James Bergstra JAMES.BERGSTRA@UMONTREAL.CA
Yoshua Bengio YOSHUA.BENGIO@UMONTREAL.CA

Hyperopt: A Python Library for Optimizing the Hyperparameters of Machine Learning Algorithms

James Bergstra*†, Dan Yamins‡, David D. Cox§

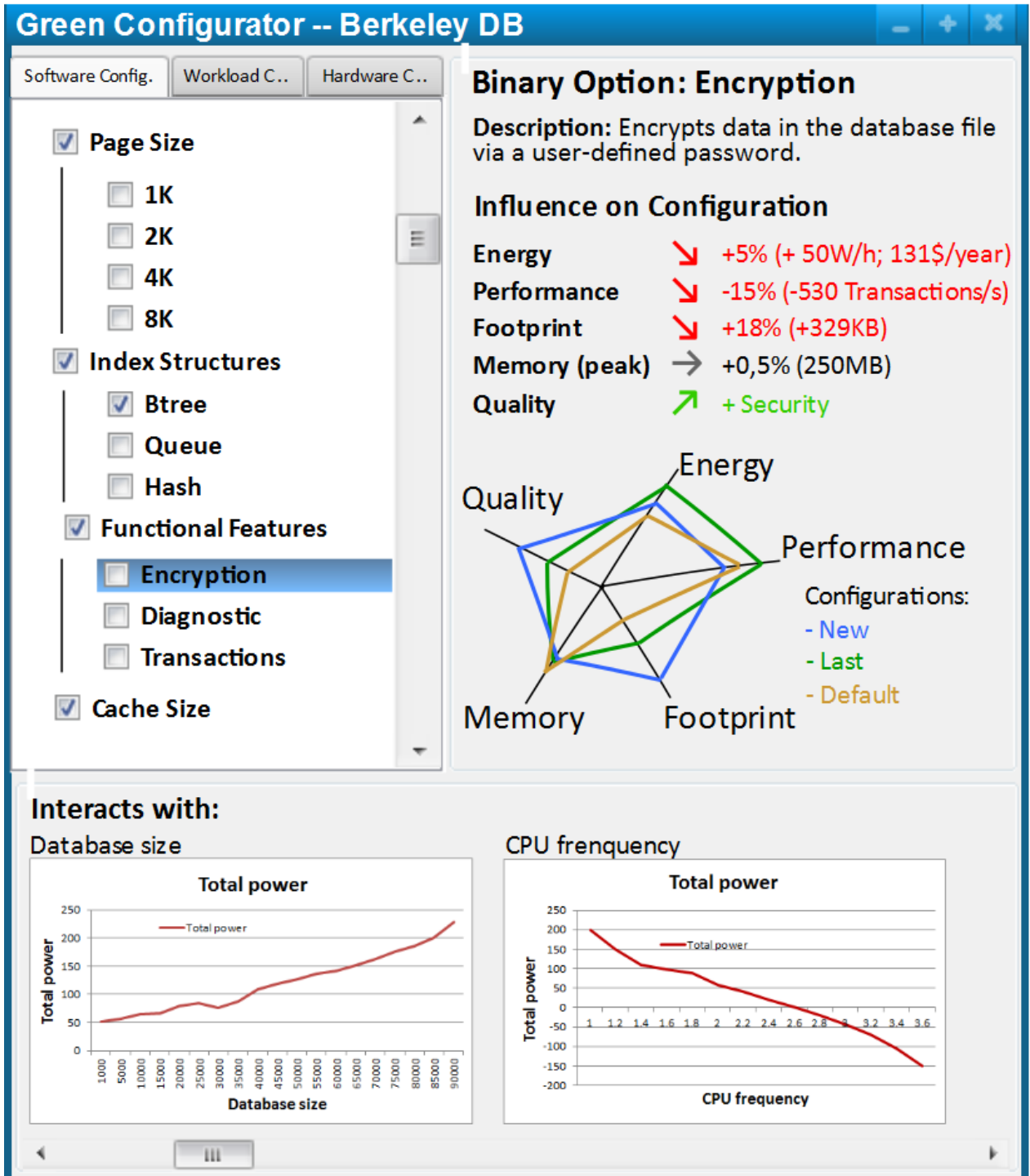
Gaussian processes to model uncertainty



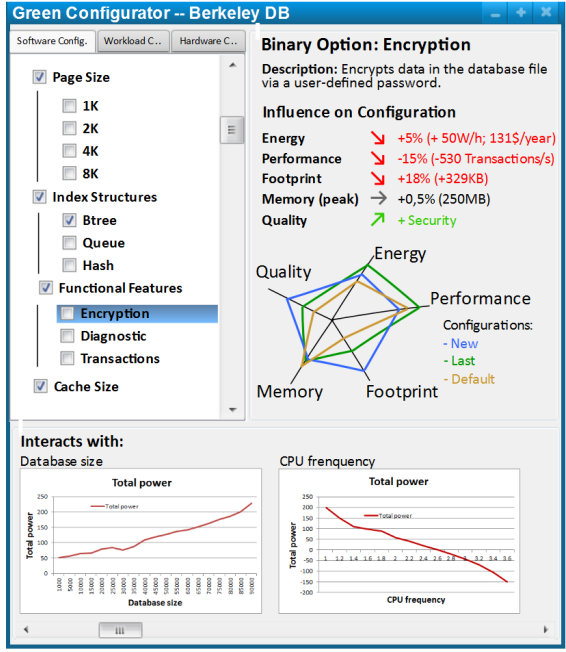


Model Inference

Configuration



Model Inference

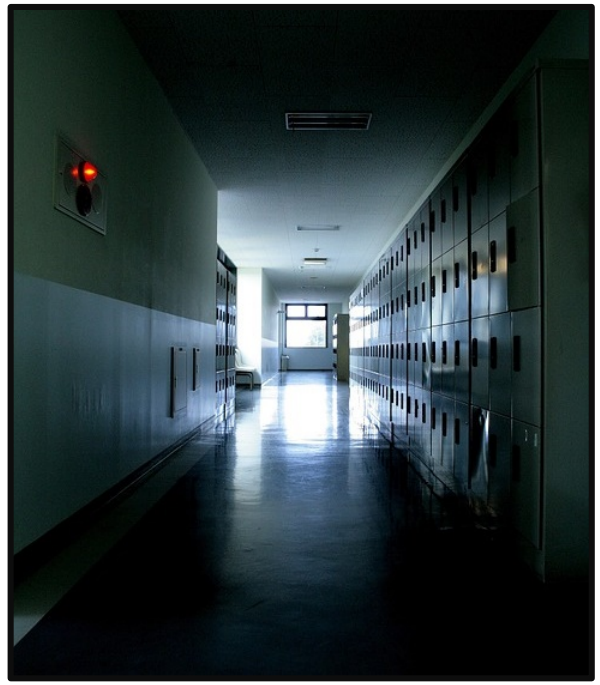


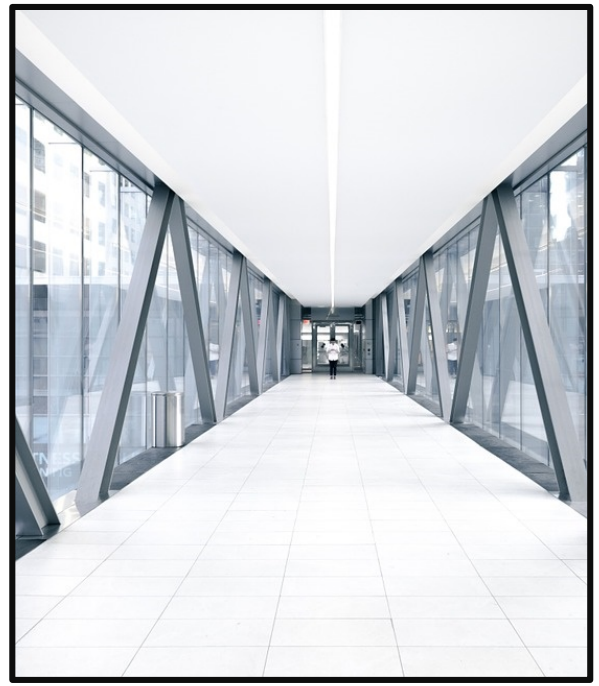
Goal: Predict configuration's performance or
explain performance influence of options

Given implementation, use cases: Understanding, debugging, planning, …

Typically sampling + learning, various techniques

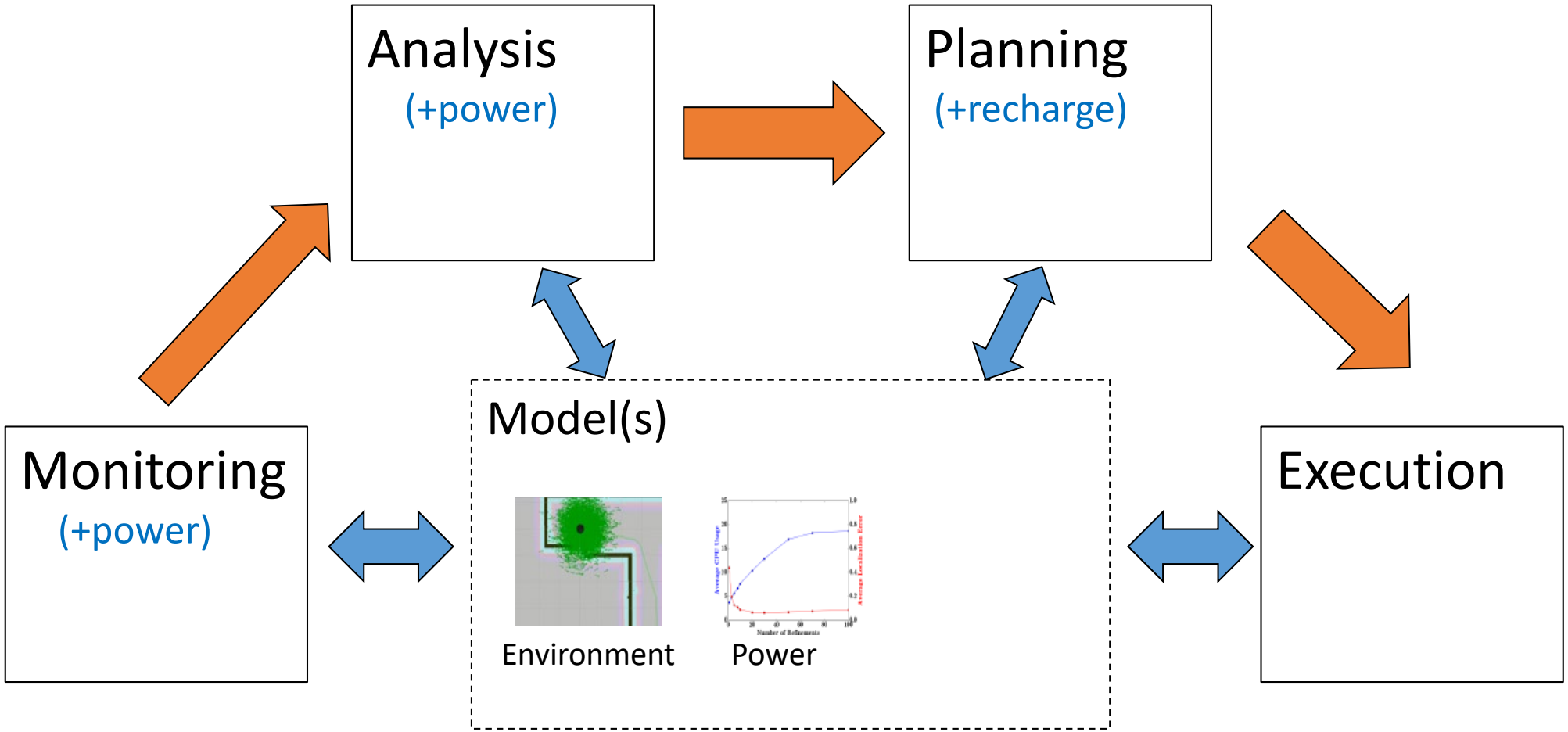
Evaluated by prediction accuracy



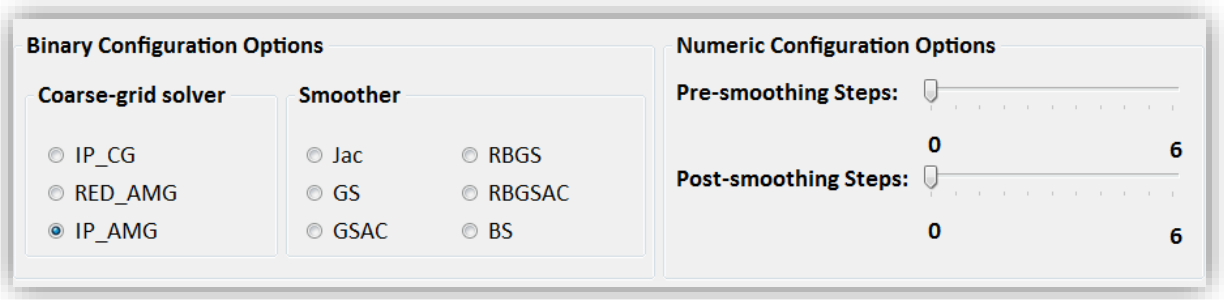


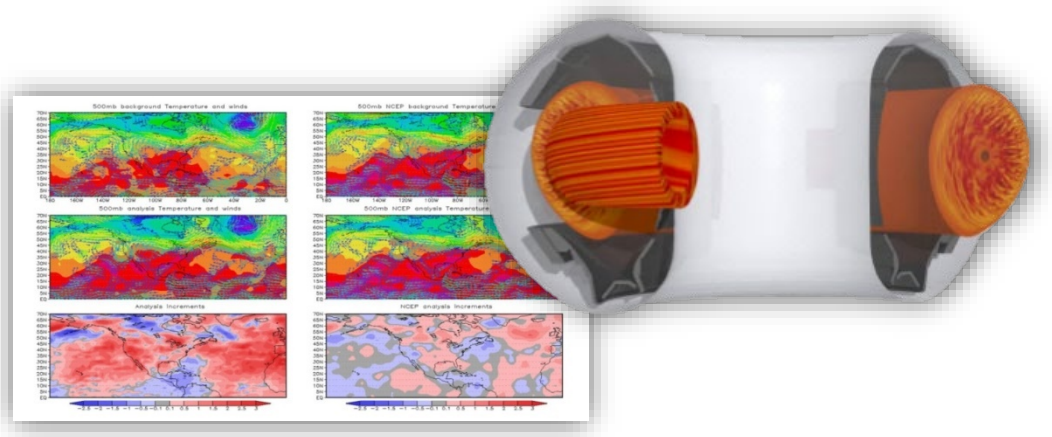

irobot

Planning


Analysis
(+power)
Planning
(+recharge)
Monitoring
(+power)
Model(s)
Environment
Power
Execution

Debugging



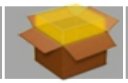


"the HIPA^{CC} expert was surprised to see that *pixelsPerThread* configuration option had only a small influence on system performance"

Sensitivity Analysis

One-At-A-Time Sensitivity Analysis

Options

					Perf.
1	0	0	1	0	10.3

Configurations

Random Sampling + Regression

Individual Options

Configurations

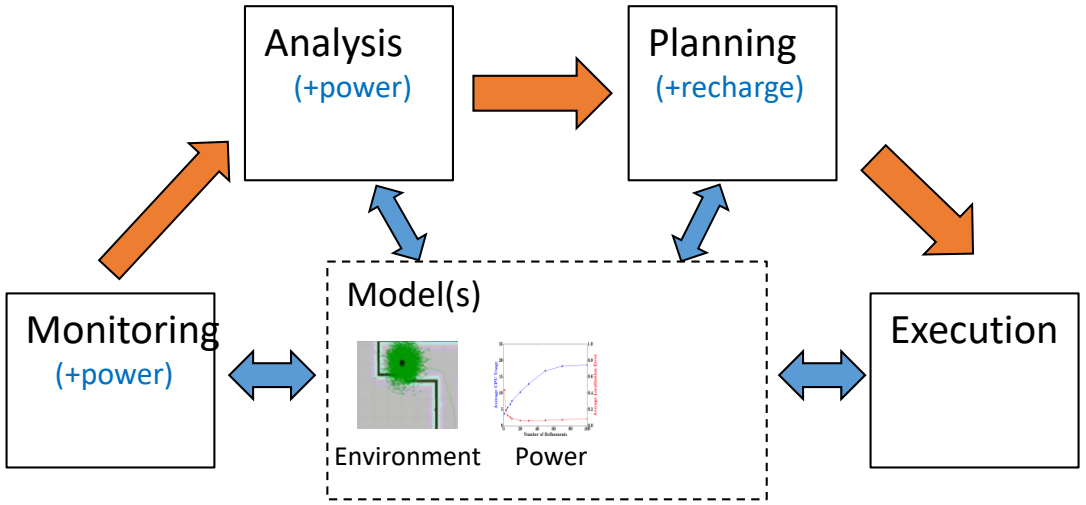
					Perf.
1	0	0	1	0	10.3
0	1	1	0	1	55.8
1	0	1	1	1	21.9
1	1	0	1	1	60.1
1	1	1	0	0	59.3
0	0	0	1	0	29.3
1	1	1	1	1	45.6

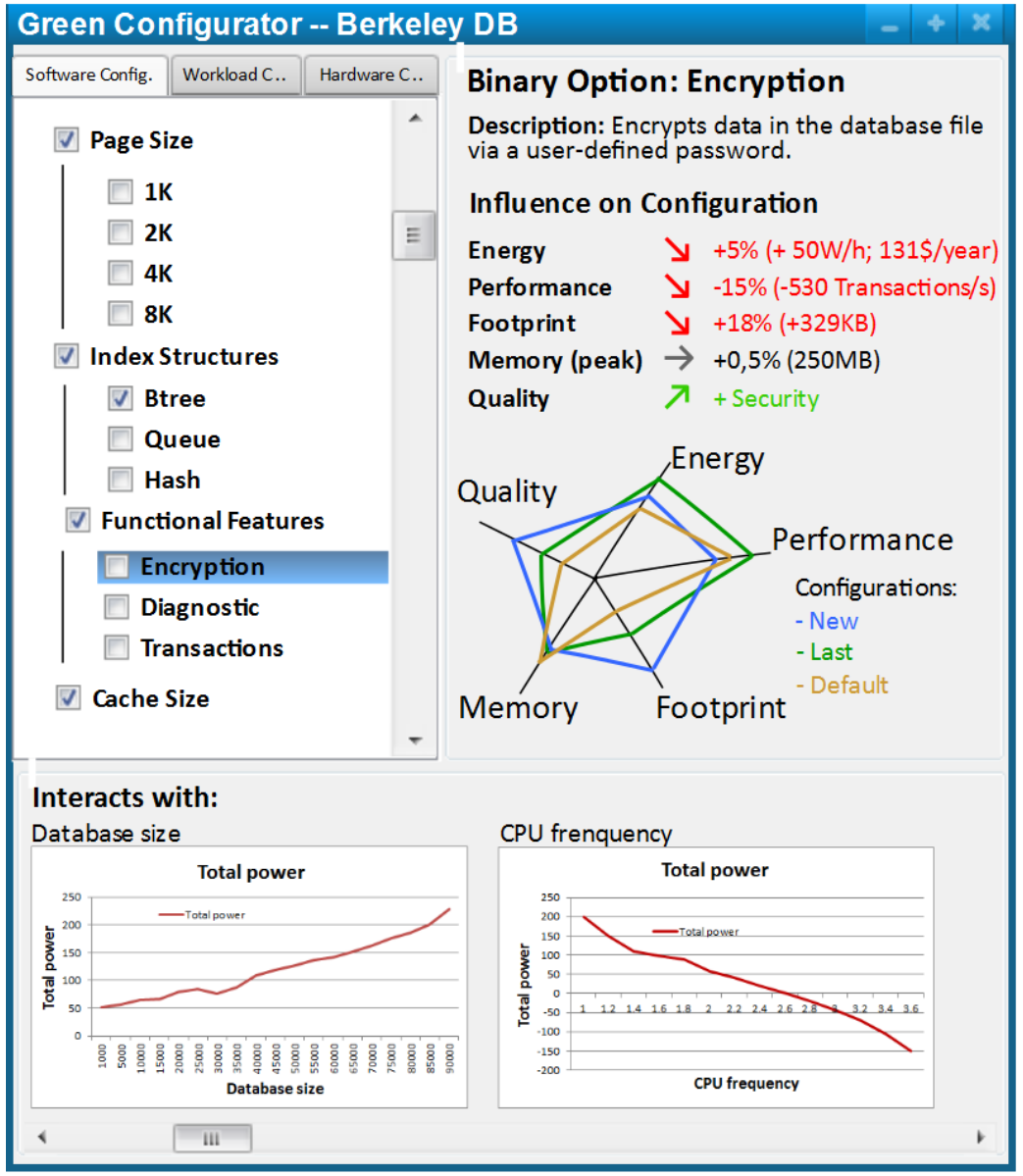
β_0	β_1	β_2	β_3	β_4	β_5	Error:
36.5	-24.3	15.3	-2.5	5.4	0.3	21%

$$p = \beta_0 + \beta_1 * index + \beta_2 * encr + \beta_3 * compr + \ldots$$

SPL Conqueror

[Siegmund,FSE'15]





$$p = 36.5 - 24.3*index + 15.3*encr - 2.5*compr + \ldots$$

Interactions?

Benchmark execution: 30 sec

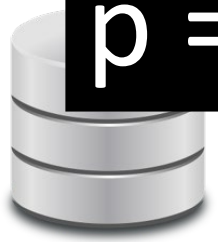
Benchmark execution: 10 sec

Index reduces by 20 sec

Benchmark execution: 35 sec

Encryption adds 5 sec

$$p = 30 - 20*index + 5*encr$$



Benchmark execution: 30 sec

+ Index

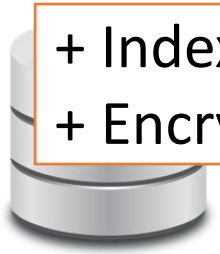
Benchmark execution: 10 sec

Index reduces by 20 sec

+ Encryption

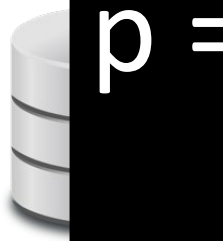
Benchmark execution: 35 sec

Encryption adds 5 sec



+ Index
+ Encryption

Benchmark execution: 50 sec



$$p = 30 - 20*index + 5*encr + 25*index*encr + \ldots$$

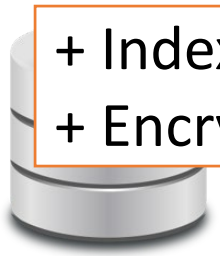
ec

Benchmark execution: 10 sec

Index reduces by 20 sec

Benchmark execution: 35 sec

Encryption adds 5 sec



Benchmark execution: 50 sec

Random Sampling + Regression

Configurations	Individual Options					Interactions			Perf.
	1	0	0	1	0	0	0	0	10.3
	0	1	1	0	1	0	0	0	55.8
	1	0	1	1	1	0	0	0	21.9
	1	1	0	1	1	1	0	1	60.1
	1	1	1	0	0	1	1	0	59.3
	0	0	0	1	0	0	0	0	29.3
	1	1	1	1	1	1	1	1	45.6
	β_0	β_1	β_2	β_3	β_4	β_5	β_6	β_7	β_8	Error:
	36.5	-24.3	15.3	-2.5	5.4	0.3				21%
	30.4	-21.1	5.3	-2.4	3.5	0.2	26.1	-5.4	0.01	6%

$$p = \beta_0 + \beta_1 * index + \beta_2 * encr + \beta_3 * compr + \ldots$$

SPL Conqueror

[Siegmund,FSE'15]

Random Sampling + Regression

Individual Options

					2	Perf.
1	0	0	1	0	1	10.3
0	1	1	0	1	0	55.8
1	0	1	1	1	1	21.9
1	1	0	1	1	1	60.1
2	1	1	0	0	4	59.3
0	0	0	1	0	0	29.3
1	1	1	1	1	1	45.6
β_0	β_1	β_2	β_3	β_4	β_5	β_6	Error:

Configurations

$$p = \beta_0 + \beta_1 * index + \ldots + \beta_6 * index^2$$

SPL Conqueror

[Siegmund,FSE'15]

Random Sampling + Regression

Configurations	Individual Options					Interactions			Perf.
	1	0	0	1	0	0	0	0	10.3
	0	1	1	0	1	0	0	0	55.8
	1	0	1	1	1	0	0	0	21.9
	1	1	0	1	1	1	0	1	60.1
	1	1	1	0	0	1	1	0	59.3
	0	0	0	1	0	0	0	0	29.3
	1	1	1	1	1	1	1	1	45.6
	β_0	β_1	β_2	β_3	β_4	β_5	β_6	β_7	β_8	Error:
	36.5	-24.3	15.3	-2.5	5.4	0.3				21%
	30.4	-21.1	5.3	-2.4	3.5	0.2	26.1	-5.4	0.01	6%

$$p = \beta_0 + \beta_1 * index + \beta_2 * encr + \beta_3 * compr + \ldots$$

SPL Conqueror

[Siegmund,FSE'15]


320 optional, independent options
more combinations than estimated
atoms in the universe

Random Sampling + Regression

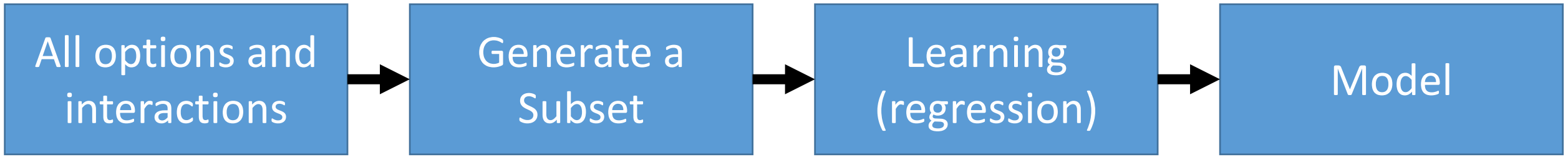
Configurations	Individual Options					Interactions			Perf.
	1	0	0	1	0	0	0	0	10.3
	0	1	1	0	1	0	0	0	55.8
	1	0	1	1	1	0	0	0	21.9
	1	1	0	1	1	1	0	1	60.1
	1	1	1	0	0	1	1	0	59.3
	0	0	0	1	0	0	0	0	29.3
	1	1	1	1	1	1	1	1	45.6
	β_0	β_1	β_2	β_3	β_4	β_5	β_6	β_7	β_8	Error:
	36.5	-24.3	15.3	-2.5	5.4	0.3				21%
	30.4	-21.1	5.3	-2.4	3.5	0.2	26.1	-5.4	0.01	6%

$$p = \beta_0 + \beta_1 * index + \beta_2 * encr + \beta_3 * compr + \ldots$$

SPL Conqueror

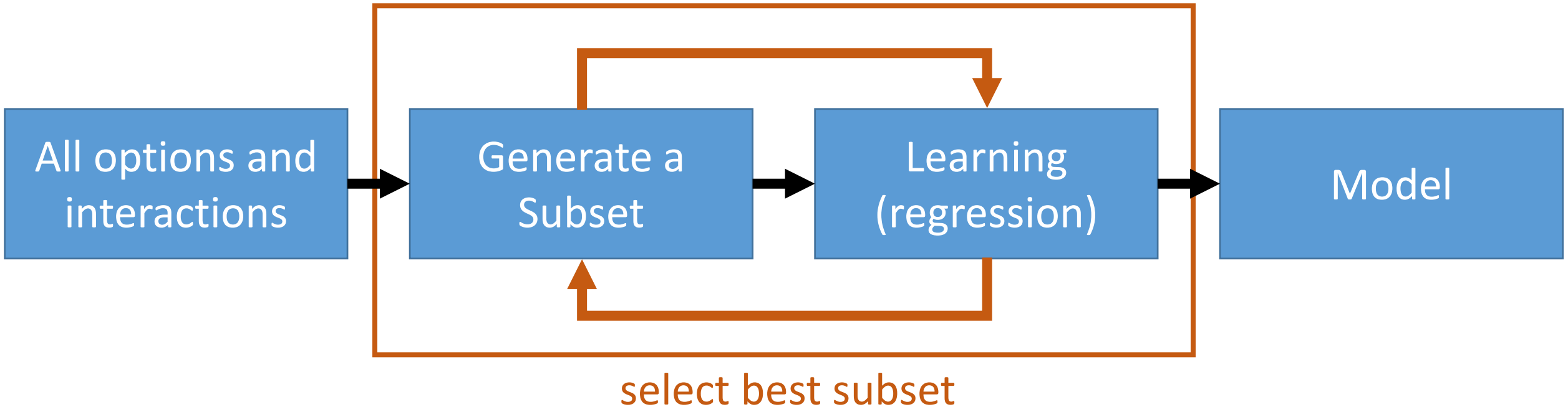
[Siegmund,FSE'15]

Feature Selection



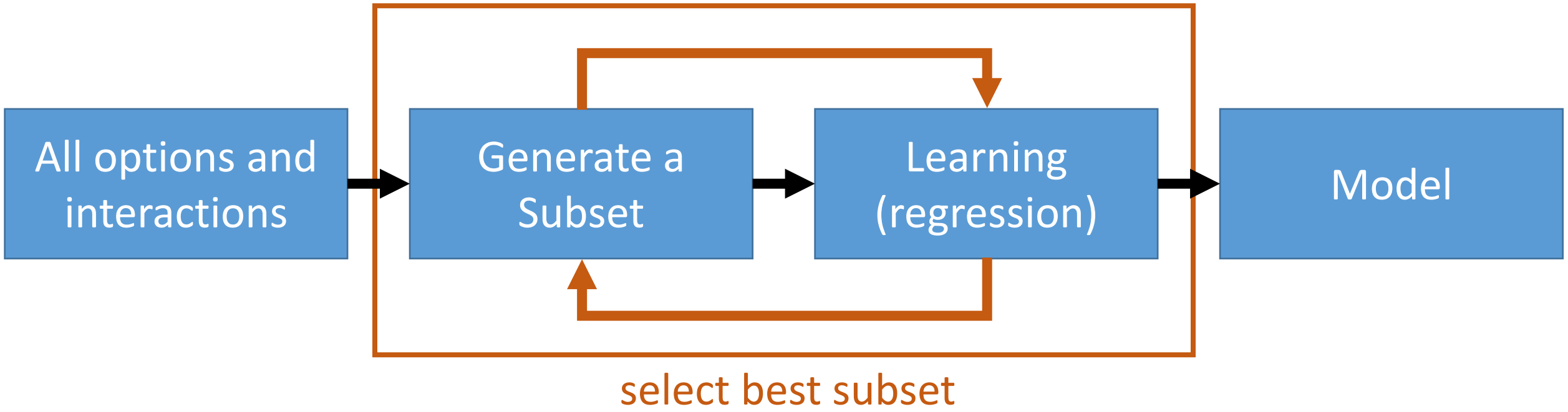
Goal: Select small set of features that best explain observations

Feature Selection



Goal: Select small set of features that best explain observations

Feature Selection



heuristics: try individual options first, then combinations of selected options

Goal: Select small set of features that best explain observations

Round 1

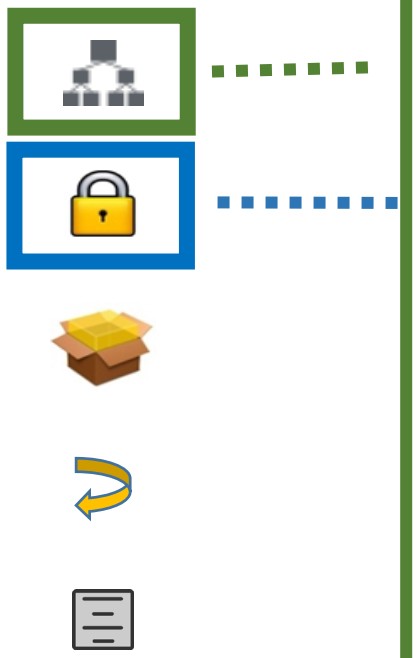
Round 1

	Perf.
1	10.3
0	55.8
1	21.9
1	60.1
1	59.3
0	29.3
1	45.6

β_0 β_1 Error:

$p = 36.5 - 24.3*index$

Round 1



	Perf.
1	10.3
0	55.8
1	21.9
1	60.1
1	59.3
0	29.3
1	45.6

β_0 β_1 Error:

$p = 36.5 - 24.3*index$

	Perf.
0	10.3
1	55.8
0	21.9
1	60.1
1	59.3
0	29.3
1	45.6

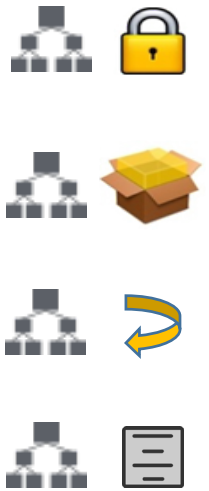
β_0 β_2 Error:

$p = 26.5 + 8.4*encr$

Round 1

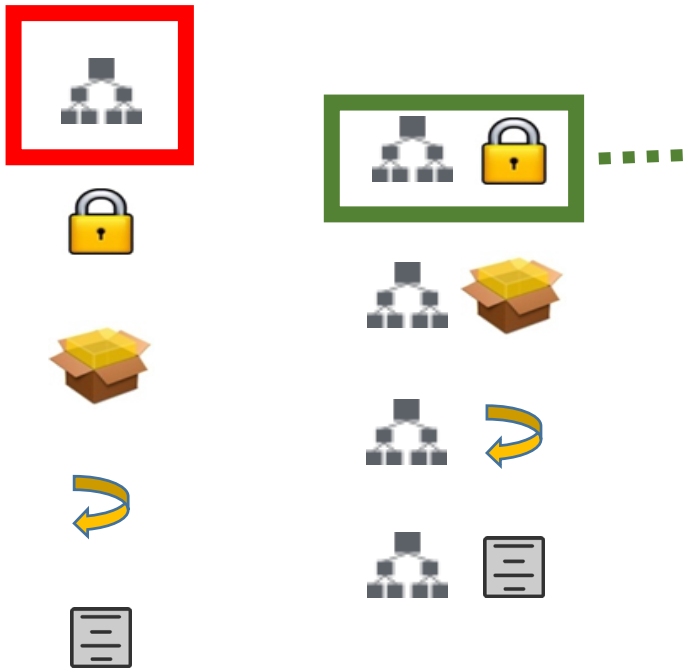
Round 1

Round 2



Round 1

Round 2



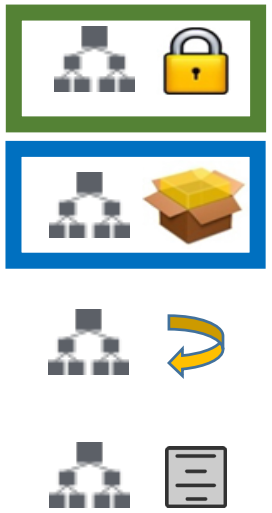
		Perf.
1	0	10.3
0	1	55.8
1	0	21.9
1	1	60.1
1	1	59.3
0	0	29.3
1	1	45.6

β_0 β_1 β_2 Error:

$$p = 32.5 - 19.3*i + \ldots$$

Round 1

Round 2



		Perf.
1	0	10.3
0	1	55.8
1	0	21.9
1	1	60.1
1	1	59.3
0	0	29.3
1	1	45.6

β_0 β_1 β_2 Error:

$p = 32.5 – 19.3*i + \ldots$

		Perf.
1	0	10.3
0	1	55.8
1	0	21.9
1	1	60.1
1	1	59.3
0	0	29.3
1	1	45.6

β_0 β_1 β_2 Error:

$p = 26.5 + 8.4*enc - \ldots$

Round 1

Round 2

Round 3

Round 4

heuristics: try individual options first, then combinations of selected options

3.55·ref + 0.01·keyint + 0.05·bframes

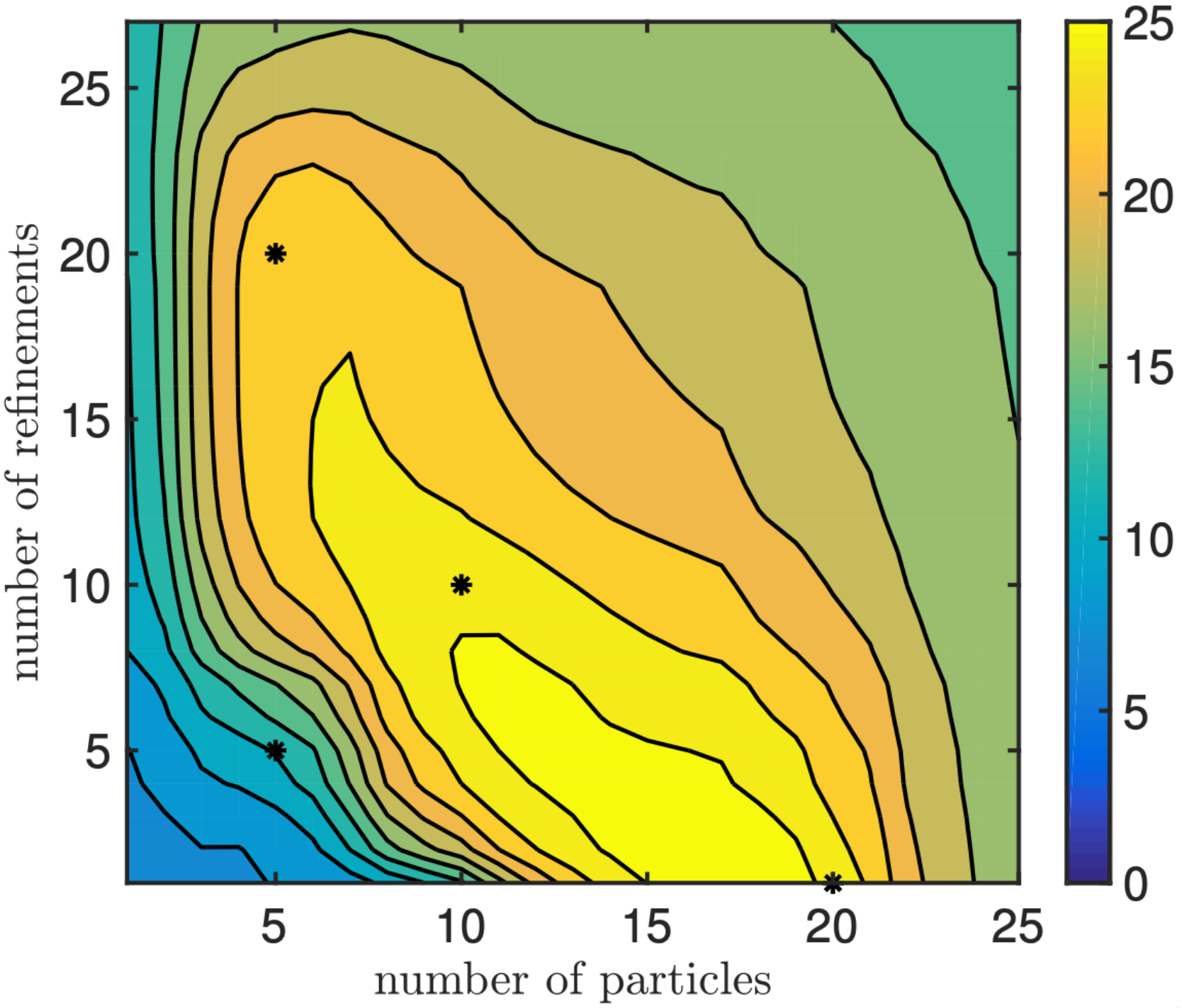
1.87·ref + 0.10·keyint + 0.5·bframes + -
0.63·ref·no_asm + 0.19·keyint·no_asm + -
0.02·bframes·crfRatio + 1.8E-
0.8·keyint·rc_lookahead·rc_lookahead +
0.0003·bframes·crfRatio·crfRatio +
0.0002·qpSetting·qpSetting + 20.92·no_asm + -
0.0004·no_asm·keyint·keyint + 5.80E-
07·bframes·keyint·keyint + -0.50·no_asm·crfRatio

Empirical Observations

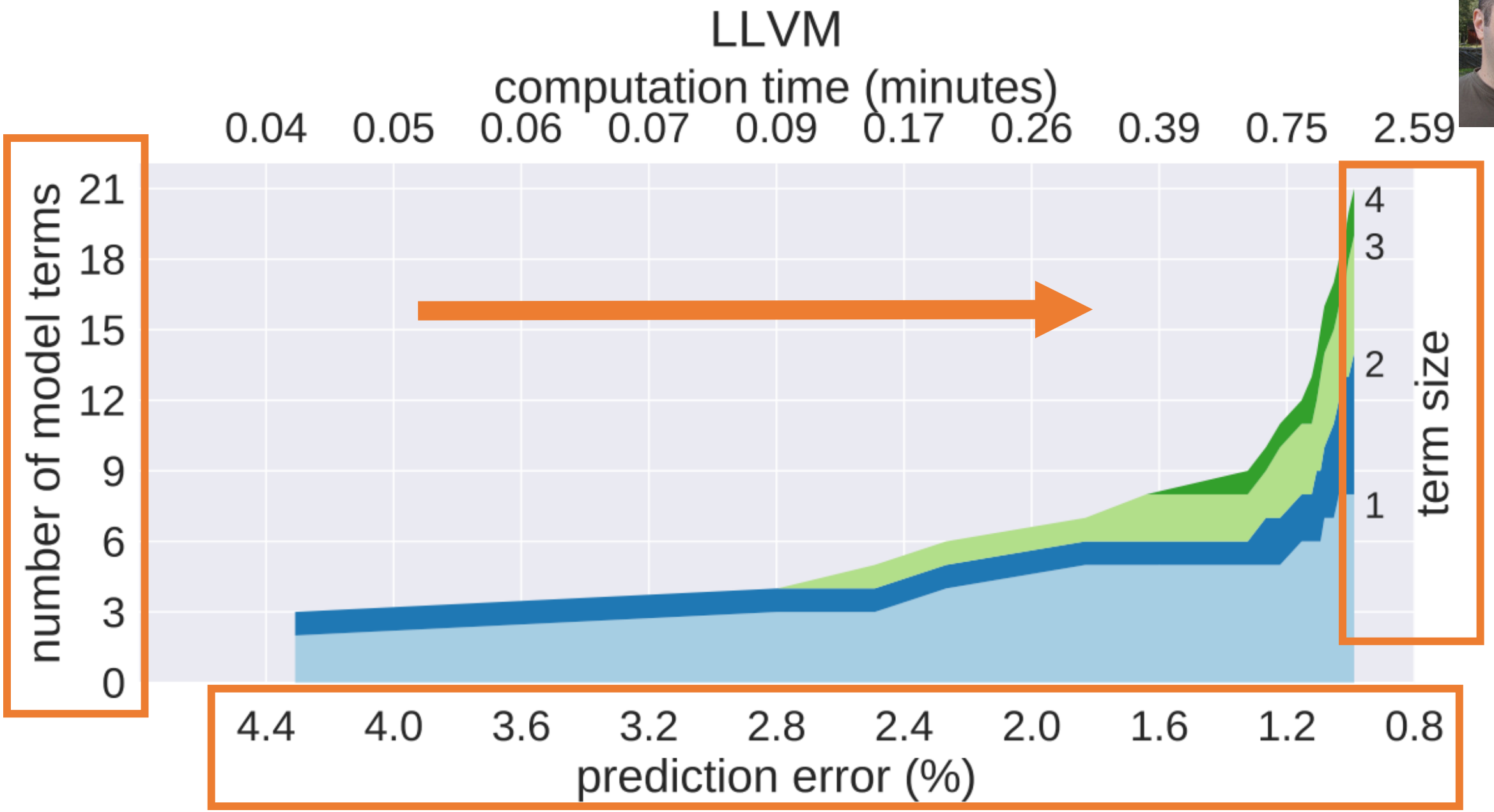
Few options responsible for most of the variation

Interactions are important to explain variation

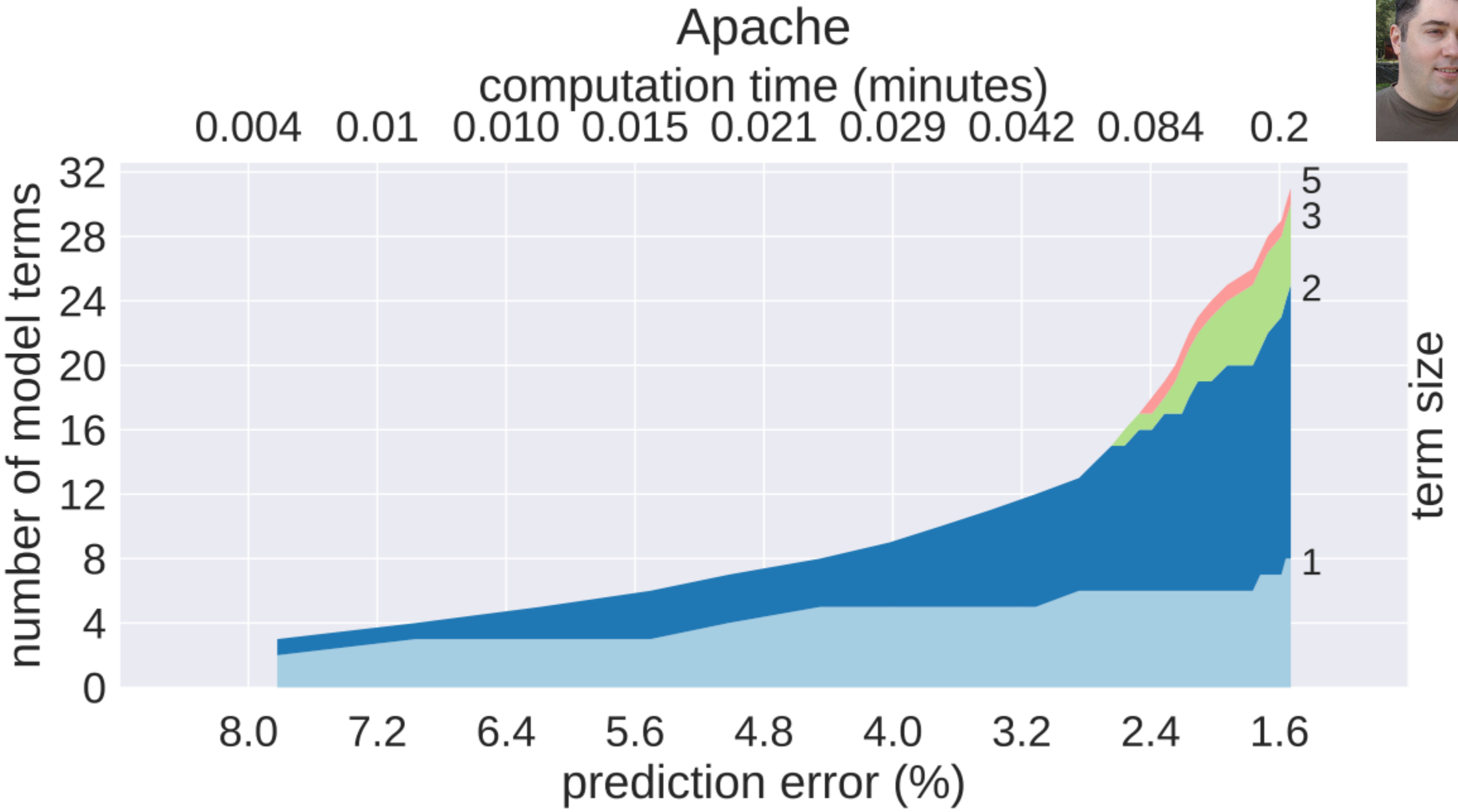
Few options interact



Performance behavior has "structure"


LLVM
computation time (minutes)
0.04
0.05
0.06
0.07
0.09
0.17
0.26
0.39
0.75
2.59
number of model terms
21
18
15
12
9
6
3
0
term size
4
3
2
1
4.4
4.0
3.6
3.2
2.8
2.4
2.0
1.6
1.2
0.8
prediction error (%)

[SoSym'18]


Apache
computation time (minutes)
0.004
0.01
0.010
0.015
0.021
0.029
0.042
0.084
0.2
number of model terms
32
28
24
20
16
12
8
4
0
term size
5
3
2
1
8.0
7.2
6.4
5.6
4.8
4.0
3.2
2.4
1.6
prediction error (%)

[SoSym'18]

3.55·ref + 0.01·keyint + 0.05·bframes

1.87·ref + 0.10·keyint + 0.5·bframes + -
0.63·ref·no_asm + 0.19·keyint·no_asm + -
0.02·bframes·crfRatio + 1.8E-
0.8·keyint·rc_lookahead·rc_lookahead +
0.0003·bframes·crfRatio·crfRatio +
0.0002·qpSetting·qpSetting + 20.92·no_asm + -
0.0004·no_asm·keyint·keyint + 5.80E-
07·bframes·keyint·keyint + -0.50·no_asm·crfRatio

So Far: Model Inference

Sensitivity analysis: detect performance interactions from small samples

Blackbox approach, simple, quite accurate, works surprisingly well

Data and machine learning tools available

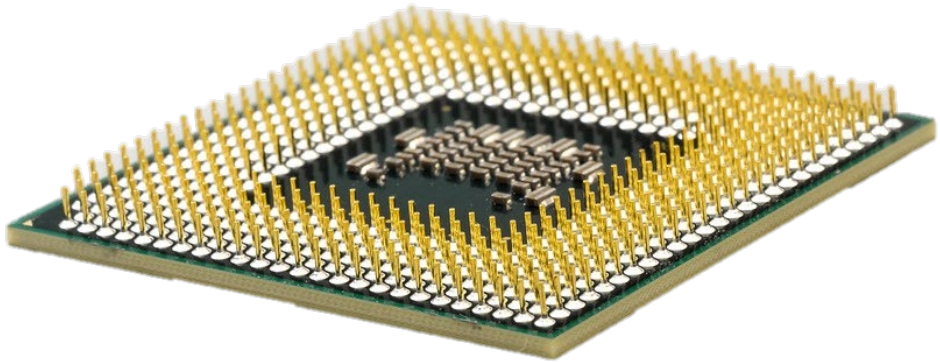
e.g., SPL Conqueror http://fosd.net/SPLConqueror

Open Challenges

Challenge: Finding Interactions

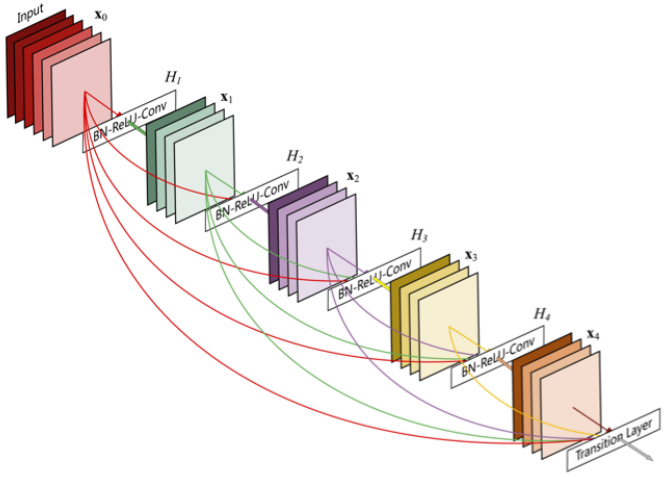
Challenge: Measurement Costs

Challenge: Workload/Environment Changes



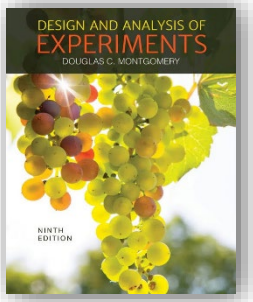
Recent Advances

Better Learning



Various learning approaches explored:
linear regression, Gaussian processes, decision trees, DNN, …
+ tuning of those

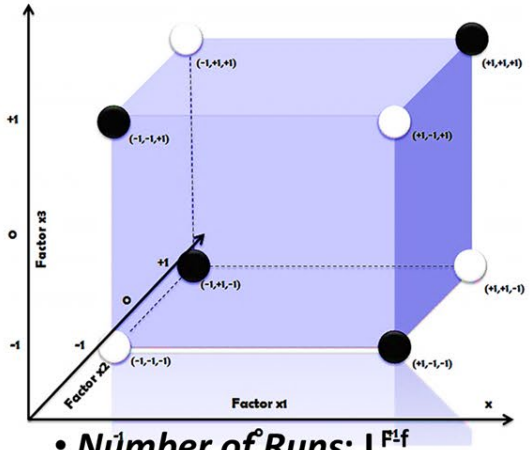
Better Sampling



PLACKETTE BURMAN

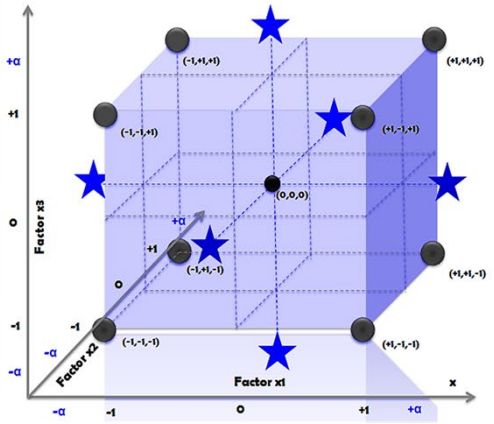
	X_1	X_2	X_3	X_4	X_5	X_6	X_7	X_8	X_9	X_{10}	X_{11}
1	+1	+1	+1	+1	+1	+1	+1	+1	+1	+1	+1
2	-1	+1	-1	+1	+1	+1	-1	-1	-1	+1	-1
3	-1	-1	+1	-1	+1	+1	+1	-1	-1	-1	+1
4	+1	-1	-1	+1	-1	+1	+1	+1	-1	-1	-1
5	-1	+1	-1	-1	+1	-1	+1	+1	+1	-1	-1
6	-1	-1	+1	-1	-1	+1	-1	+1	+1	+1	-1
7	-1	-1	-1	+1	-1	-1	+1	-1	+1	+1	+1
8	+1	-1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

FRACTIONAL FACTORIAL



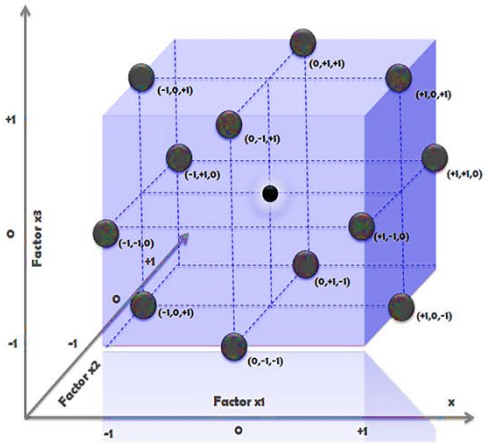
- *Number of Runs*: L^{F-f}

CENTRAL COMPOSITE



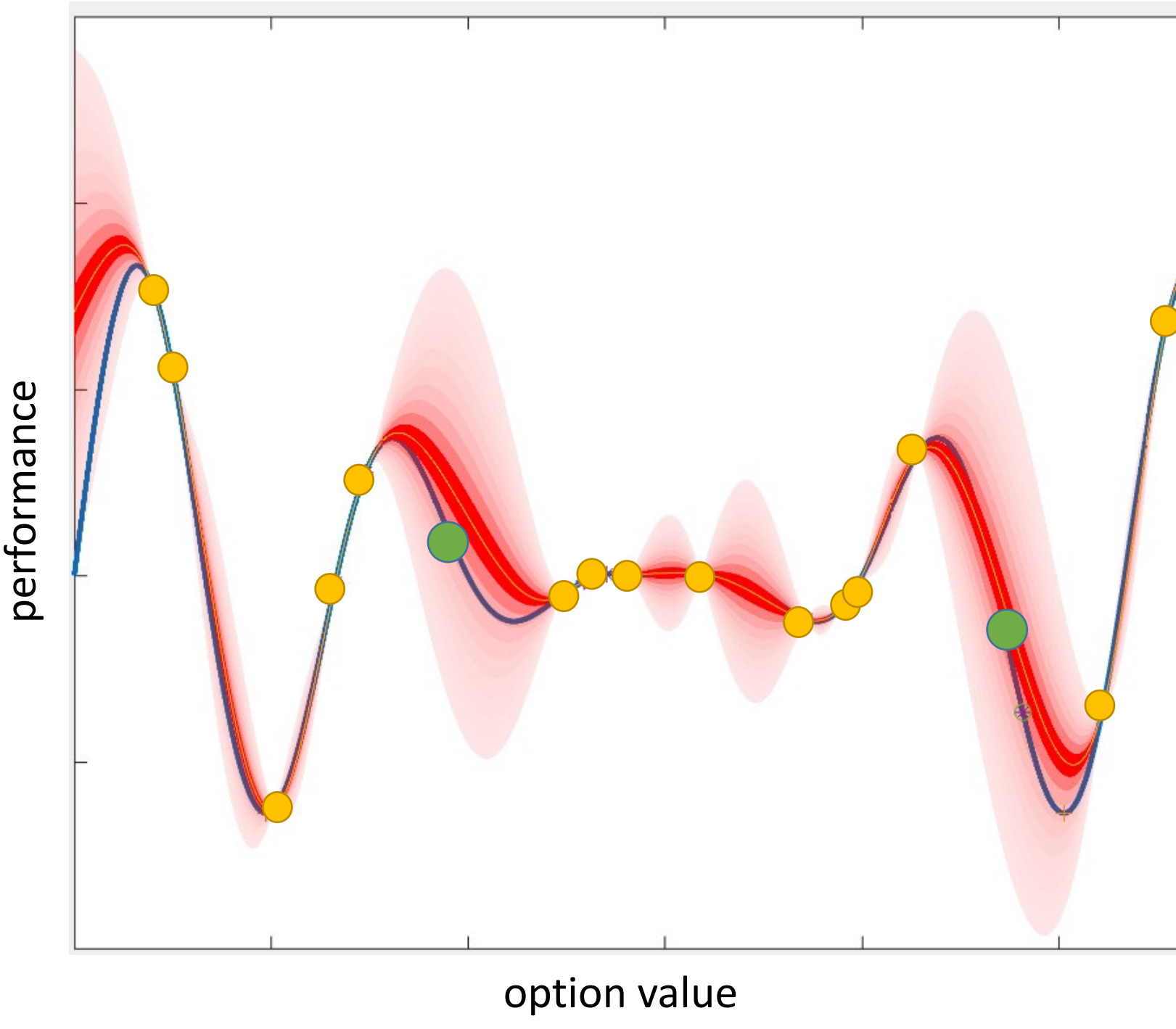
- ***Levels:*** **5**
  [‘‘- **α**’,`-1' ‘0’ and `+1',’ **+ α**’]
- ***No of Runs:*** 2^{fp} **+ 2SP + CP**

BOX BEHNKEN



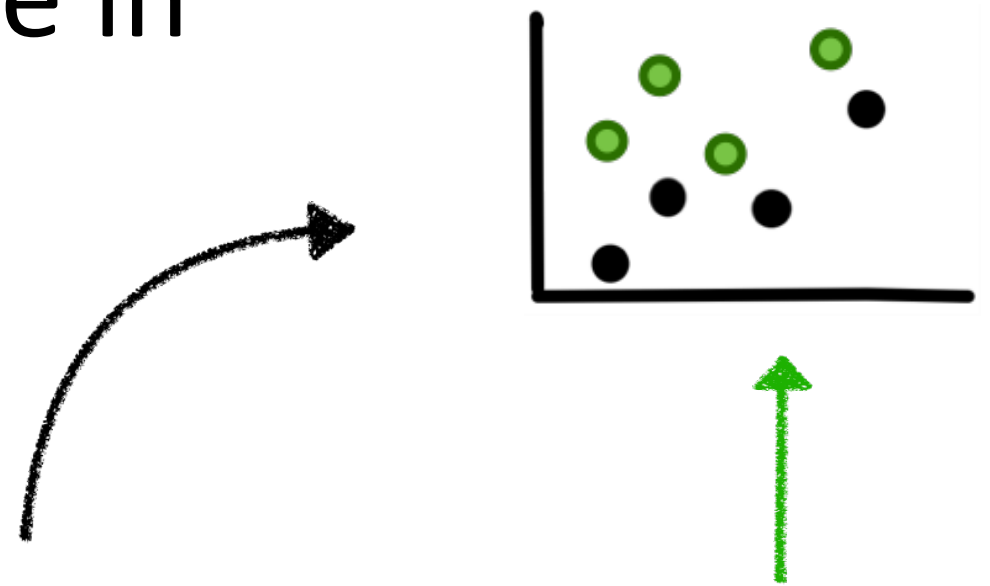
- ***Levels:*** **3** levels per factor
- ***Design Points:*** at the “**mid” points**
  edges of the process space & center

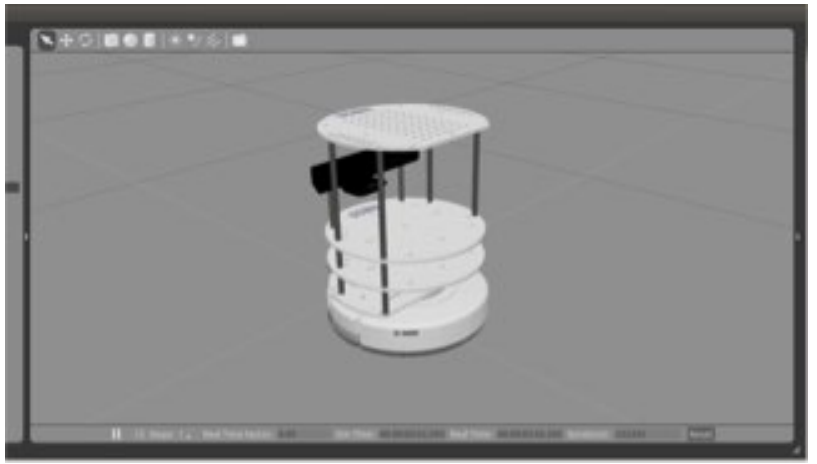
Active Sampling



Reducing Measurement Costs

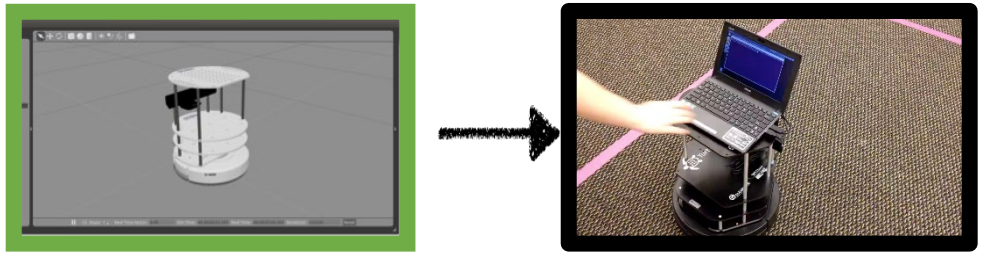
Performance of real system is “**similar**” to performance in simulators

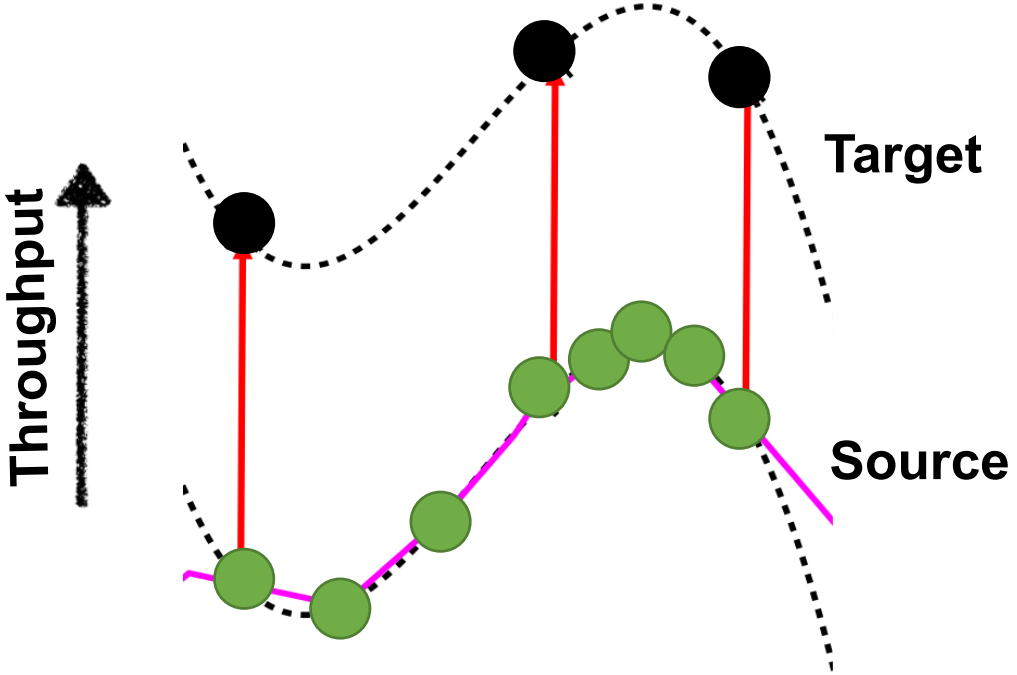




[SEAMS’17]

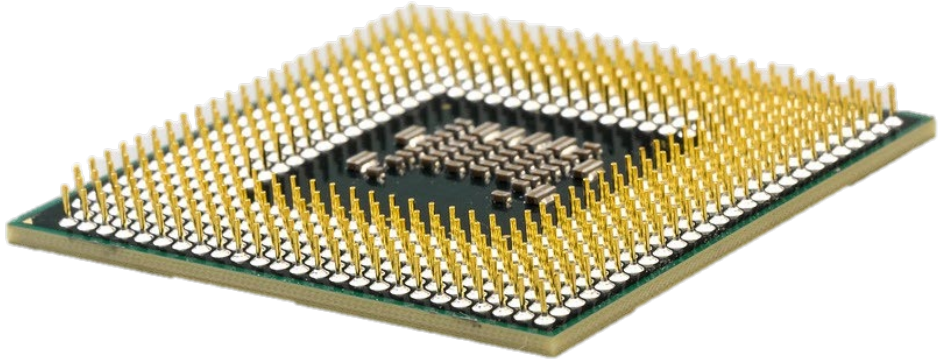
Transfer Learning





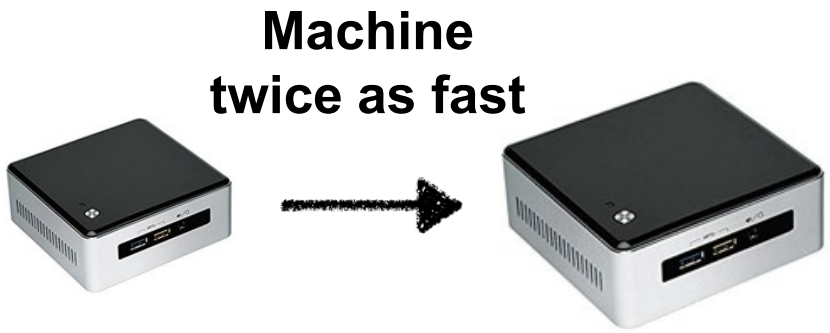
Workload and Hardware Changes

Most approaches assume fixed workload, learned models specific to benchmark



Model workload and hardware variability as additional options

Transfer Learning



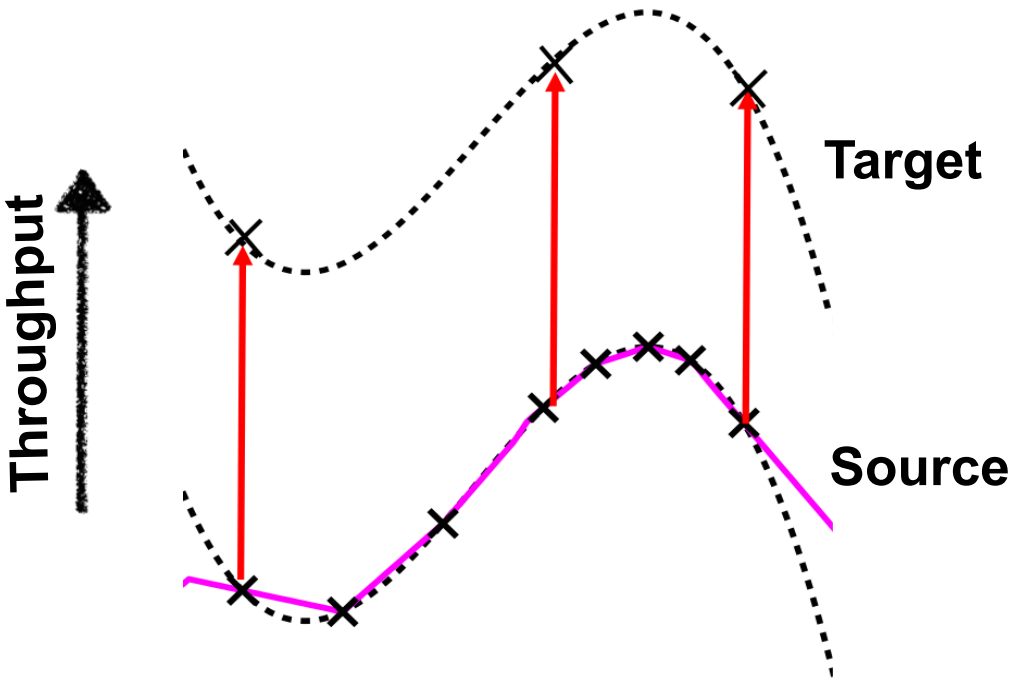


TABLE II: Results indicate that there exist several forms of knowledge that can be transfered across environments and can be used in transfer learning.

		RQ1					RQ2					RQ3				RQ4			
		H1.1	H1.2	H1.3	H1.4		H2.1				H2.2	H3.1			H3.2	H4.1			H4.2
Environment	ES	M1	M2	M3	M4	M5	M6	M7	M8	M9	M10	M11	M12	M13	M14	M15	M16	M17	M18
SPEAR— Workload (#variables/#clauses): $w_1 : 774/5934, w_2 : 1008/7728, w_3 : 1554/11914, w_4 : 978/7498$; Version: $v_1 : 1.2, v_2 : 2.7$																			
$ec_1 : [h_2 \rightarrow h_1, w_1, v_2]$	S	**1.00**	0.22	**0.97**	**0.92**	**0.92**	9	7	7	0	1	25	25	25	1.00	0.47	0.45	1	1.00
$ec_2 : [h_4 \rightarrow h_1, w_1, v_2]$	L	0.59	24.88	**0.91**	**0.76**	**0.86**	12	7	4	2	0.51	**41**	**27**	21	**0.98**	0.48	0.45	1	0.98
$ec_3 : [h_1, w_1 \rightarrow w_2, v_2]$	L	**0.96**	1.97	0.17	0.44	0.32	9	7	4	3	1	23	23	22	0.99	0.45	0.45	1	1.00
$ec_4 : [h_1, w_1 \rightarrow w_3, v_2]$	M	**0.90**	3.36	-0.08	0.30	0.11	7	7	4	3	0.99	22	23	22	0.99	0.45	0.49	1	0.94
$ec_5 : [h_1, w_1, v_2 \rightarrow v_1]$	S	0.23	**0.30**	0.35	0.28	0.32	6	5	3	1	0.32	**21**	**7**	7	0.33	0.45	0.50	1	0.96
$ec_6 : [h_1, w_1 \rightarrow w_2, v_1 \rightarrow v_2]$	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
$ec_5 : [h_1, w_3, v_2 \rightarrow$																			
$ec_6 : [h_1, w_3, v_1 \rightarrow$																			
$ec_7 : [h_1, w_1 \rightarrow w_3$																			
$ec_8 : [h_2 \rightarrow h_1, w_1$																			
SQLite— Workload																			
$ec_1 : [h_3 \rightarrow h_2, w_1,$																			
$ec_2 : [h_3 \rightarrow h_2, w_2,$																			
$ec_3 : [h_2, w_1 \rightarrow w_2, v_1]$	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
$ec_6 : [h_1, w_2 \rightarrow w_1$																			
$ec_7 : [h_2 \rightarrow h_1, w_2$																			
SaC— Workload: w																			
$ec_1 : [h_1, w_1 \rightarrow w_2$																			
$ec_2 : [h_1, w_1 \rightarrow w_3$																			
$ec_3 : [h_1, w_1 \rightarrow w_4$																			
$ec_4 : [h_1, w_1 \rightarrow w_5$																			
$ec_5 : [h_1, w_2 \rightarrow w_3$																			
$ec_6 : [h_1, w_2 \rightarrow w_4, v_1]$	S	**0.91**	5.54	**0.80**	0.00	**0.91**	14	11	8	0	0.85	64	65	31	-0.40	0.13	0.15	0.12	0.64
$ec_7 : [h_1, w_2 \rightarrow w_5$																			
$ec_8 : [h_1, w_3 \rightarrow w_4$																			
$ec_9 : [h_1, w_3 \rightarrow w_5$																			
$ec_{10} : [h_1, w_4 \rightarrow w$																			
$ec_{11} : [h_1, w_6 \rightarrow w$																			
$ec_{12} : [h_1, w_6 \rightarrow w$																			
$ec_{13} : [h_1, w_8 \rightarrow w$																			
$ec_{14} : [h_1, w_9 \rightarrow w$																			

ES: Expected severity
SaC workload descri
nbody: simulation of
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;

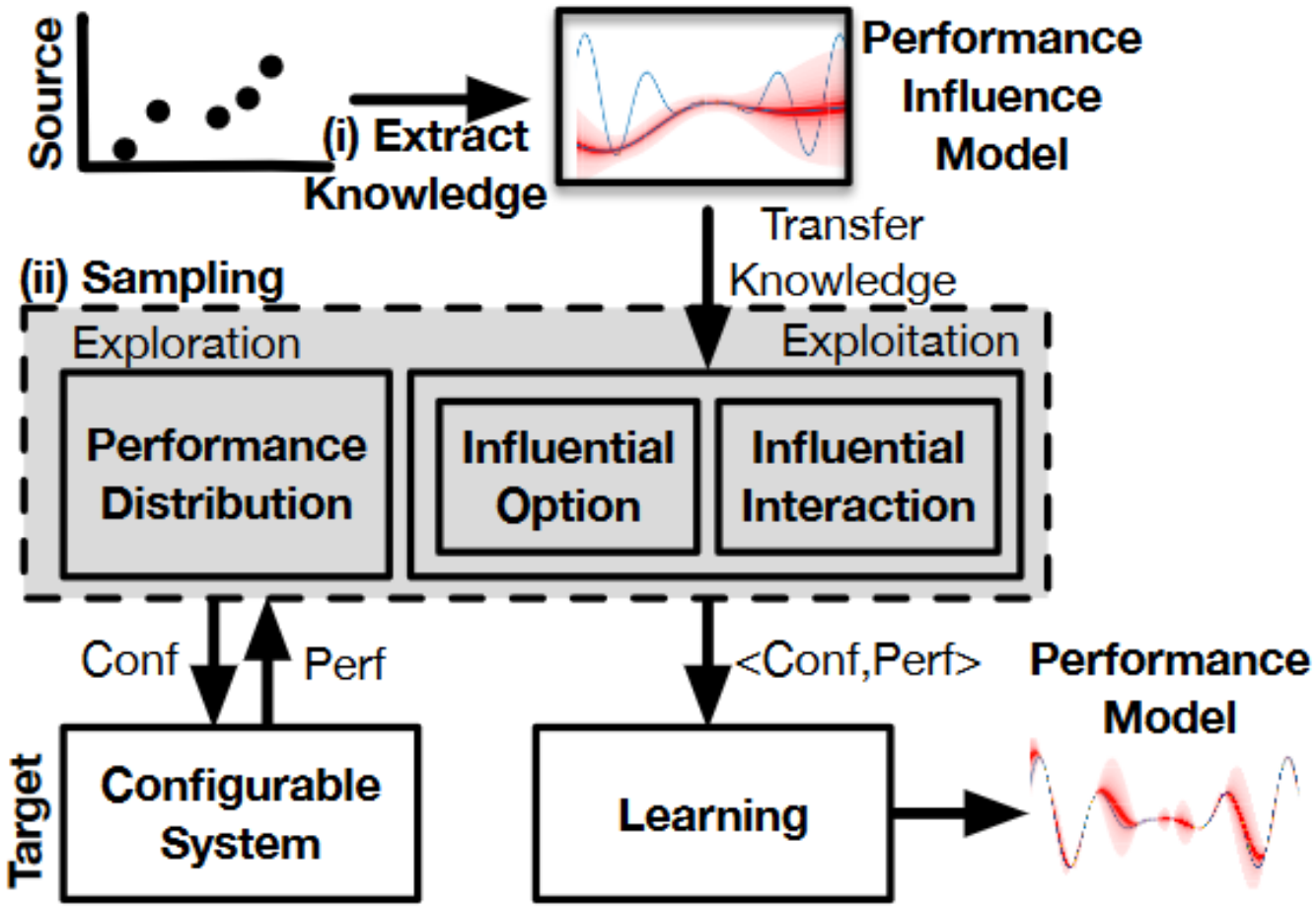
Insight 1. **Performance distributions can be transferred:** Potential for learning a non-linear transfer function.

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.

[ASE'17]

Learn to Sample

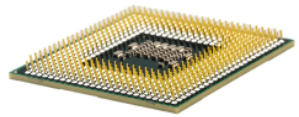


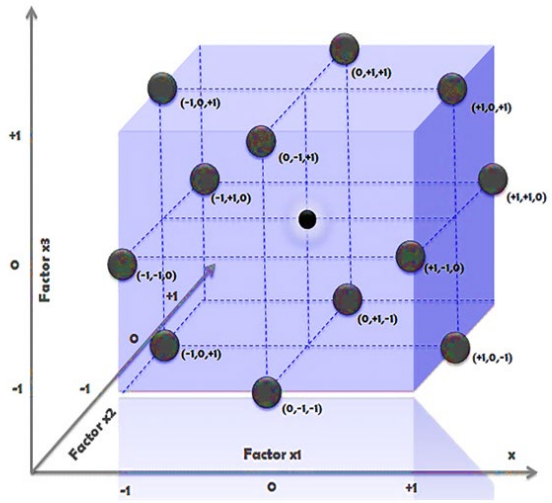
[FSE'18]

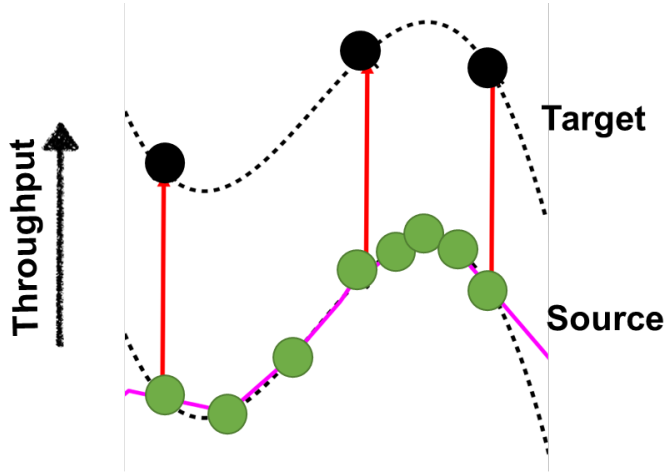
Challenge: Finding Interactions

Challenge: Measurement Costs

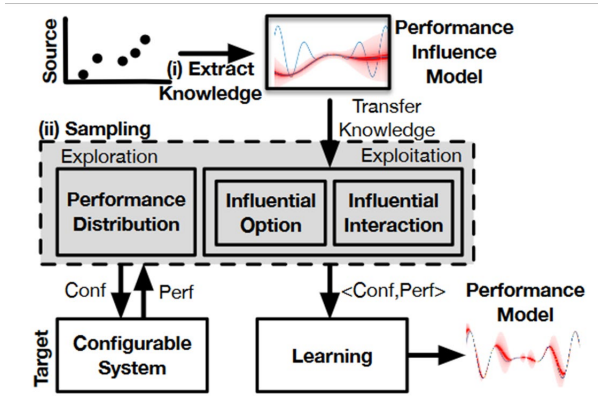
Challenge: Workload/Environment Changes







Learn to Sample



Future Directions

Exploiting Code Structure

```
#ifdef LOCKING
list *locks;
void lock() { /*...*/ }
void unlock() { /*...*/ }
#endif

void put(Object key,Object data){
#ifdef LOCKING
    lock();
#endif
    /*...*/
}

#ifdef STATISTICS
int getDbSize() {
    return calculateDbSize();
}
int calculateDbSize() {
#ifdef LOCKING
    lock();
#endif
    /*...*/
}
#endif
```

Tracking Load-Time Options

```
class ProxyService {
    static boolean NATIVE_PROXY_SUPPORTED
                    = Build.VERSION.SDK_INT >= 12;
    public void onSharedPreferenceChanged() {
        boolean enableSharing = false;
        if (!NATIVE_PROXY_SUPPORTED)
            if (Context.getSystemService("bluetooth"))
                enableSharing = true;

        if (enableSharing) {
            // slow code
        }
    }
}
```

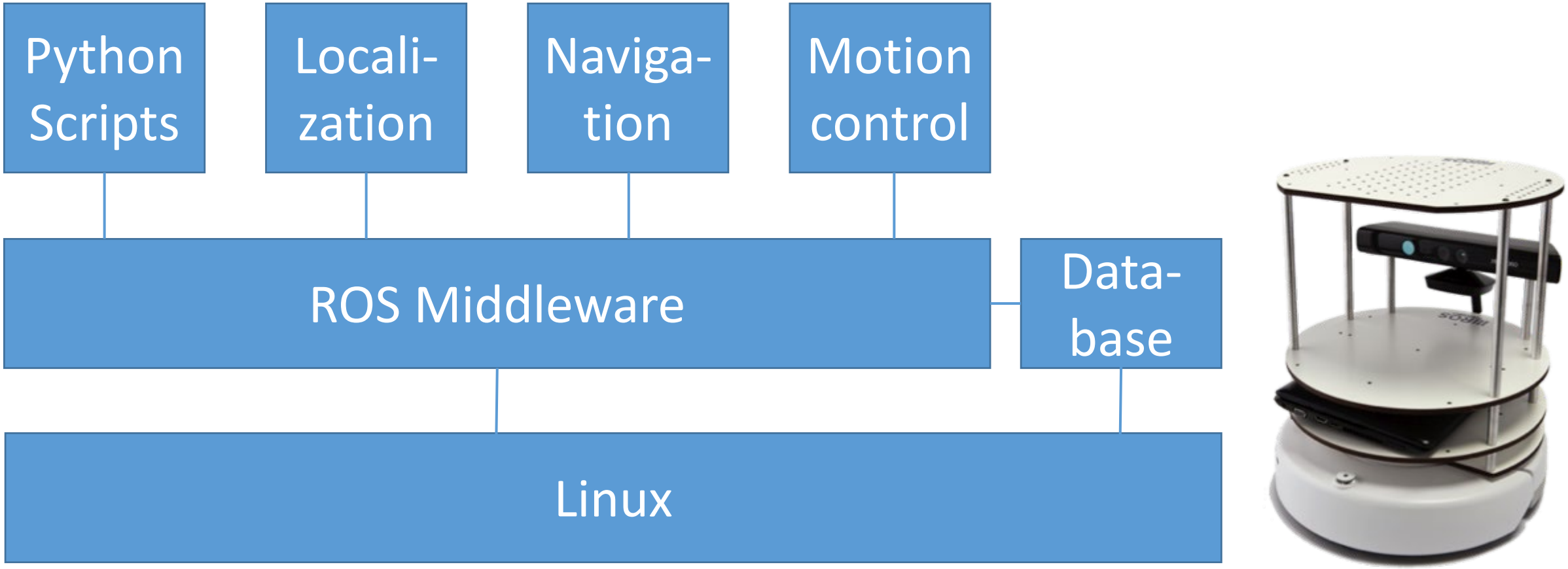
Whitebox Performance Analysis

Static/dynamic tracking of options w/ slicing & taint tracking

Which options influence performance-critical regions?
Which options interact?

-> Reduce number of measurements
-> Target measurements for likely interactions

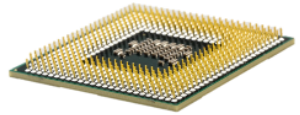
Graybox Performance Analysis?



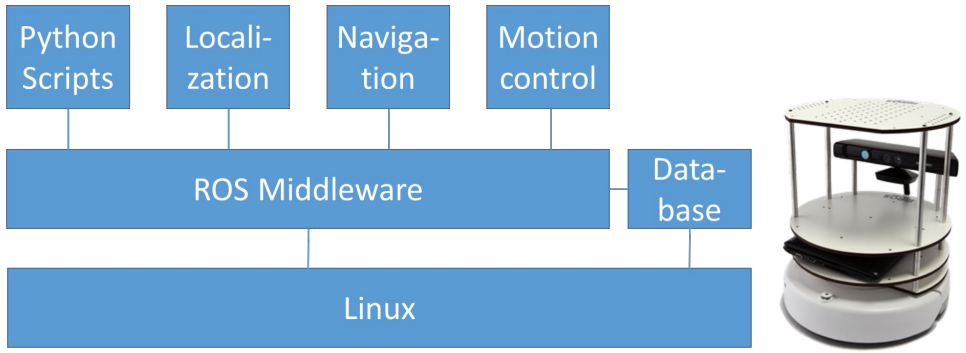
Challenge: Finding Interactions

Challenge: Measurement Costs

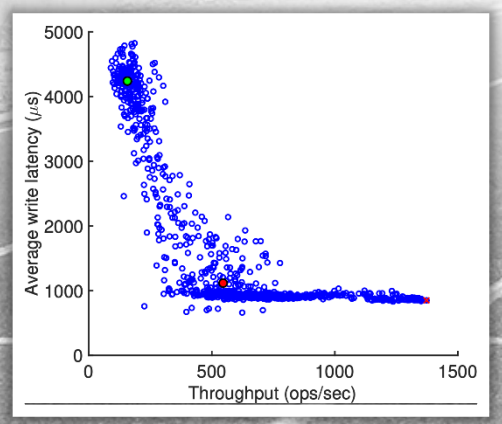
Challenge: Workload/Environment Changes

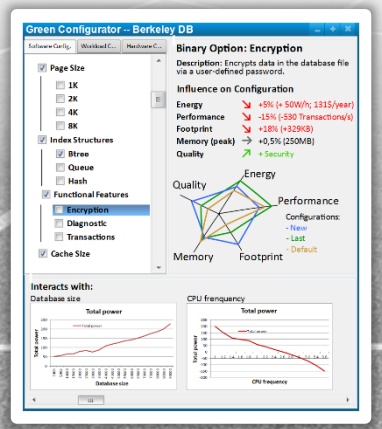


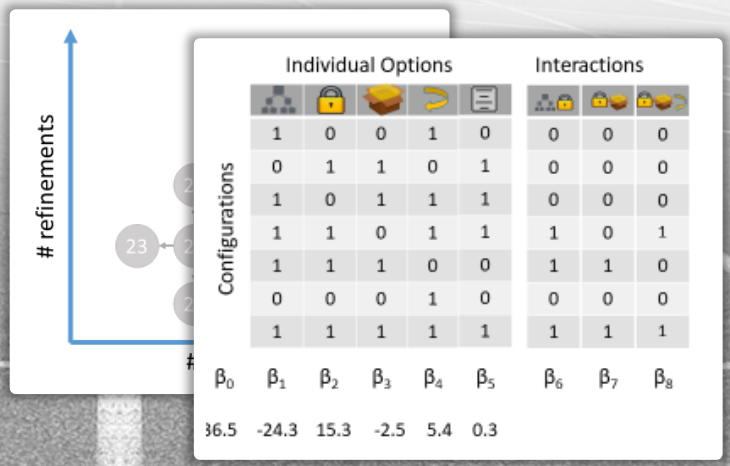
```
class ProxyService {
   static boolean NATIVE_PROXY_SUPPORTED
                = Build.VERSION.SDK_INT >= 12;
   public void onSharedPreferenceChanged() {
      boolean enableSharing = false;
      if (!NATIVE_PROXY_SUPPORTED)
         if (Context.getSystemService("bluetooth"))
           enableSharing = true;

      if (enableSharing) {
        // slow code
      }
   }
}
```

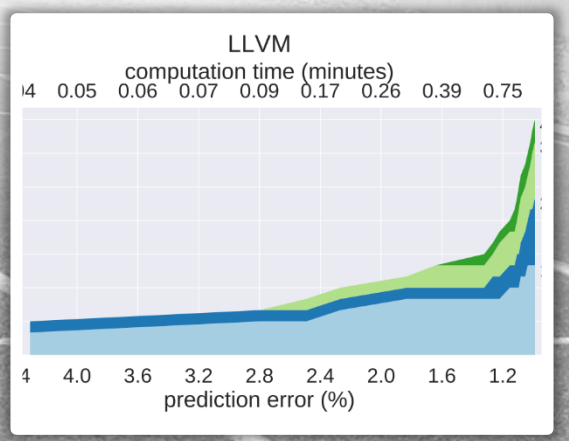


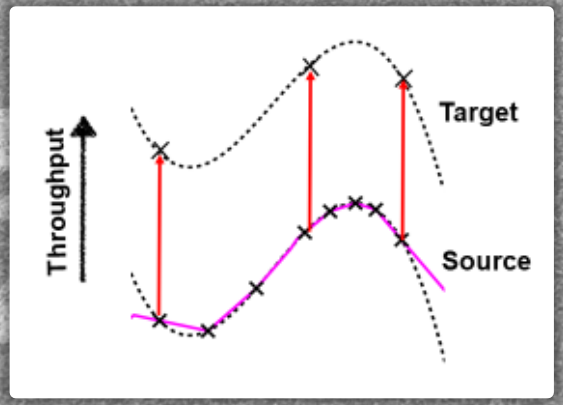
Performance Analysis for Highly-Configurable Systems












```
class ProxyService {
  static boolean NATIVE_PROXY_SUPPORTED
              = Build.VERSION.SDK_INT >= 12;
  public void onSharedPreferenceChanged() {
    boolean enableSharing = false;
    if (!NATIVE_PROXY_SUPPORTED)
      if (Context.getSystemService("bluetooth"))
        enableSharing = true;

    if (enableSharing) {
```