Frugal: Cheaper Methods for SBSE

Vivek Nair
Why configurations are so important?

- Software systems are configurable
- Configurations are parameters to control the behavior of a system
  - Configurations of Apache:
    - HostNameLookups
    - FollowSimLinks
    - ....
- Different configurations of system will result in different performance

Request

HostNameLookup = True

APACHE

Response Time = 2100 ms
Why configurations are so important?

- Software systems are configurable
- Configurations are parameters to control the behavior of a system
  - Configurations of **Apache**:
    - HostNameLookups
    - FollowSimLinks
    - ....
- Different configurations of system will result in different performance
Example

Find the fastest configuration setting for given a sample program?

Just run it?

<table>
<thead>
<tr>
<th>Conf.</th>
<th>1</th>
<th>1</th>
<th>0</th>
<th>1</th>
<th>0</th>
<th>0</th>
<th>0</th>
<th>1</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td></td>
</tr>
</tbody>
</table>
Find the fastest configuration setting for given a sample program?

Just run it?

How about now?
We need a Surrogate!

Surrogate is a cheap version of the actual system
Who endorses Surrogates?

Other Communities

- Aerospace
  - Axial compressor blade shape optimization [Samad08]
  - Hydraulic turbine diffuser shape optimization [Marjavaara07]

- Engineering Design
  - Enhanced oil recovery process [Sanchez06]
  - Design of composite materials [Sakata08]
  - Alkaline-surfactant-polymer flooding processes [Zerpa05]

Software Engineering

No surrogates....
Who endorses Surrogates?

Other Communities

- **Aerospace**
  - Axial compressor blade shape optimization [Samad08]
  - Hydraulic turbine diffuser shape optimization [Marjavaara07]

- **Engineering Design**
  - Enhanced oil recovery process [Sanchez06]
  - Design of composite materials [Sakata08]
  - Alkaline-surfactant-polymer flooding processes [Zerpa05]

Software Engineering

*No surrogates….*

Most Similar But **NOT Surrogates**:

- Heuristic method to predict response times [Siegmund’12]
- Random Sampling to build a prediction model [Guo’13, Sarkar’15]
Our Surrogate Method!

Our method “WHAT” is better than the state of the art

- Similar result using 2 to 10 times less evaluations
- Predictions are more stable
Vivek Nair, Tim Menzies, Norbert Siegmund, Sven Apel. Faster Discovery of Faster System Configurations with Spectral Learning. Submitted to FSE - 2016
BACKGROUND
“Search” in Software Engineering

What is the: [Harman’12]

- best way to structure this system to enhance its maintainability?
- smallest set of test cases that covers all branches?
- fastest configuration of this system to run this benchmark program?
Software Engineering problems are

- MultiObjective [Mkaouer’15]
  - The are more than one objective to optimize

- Multi-Modal
  - There are more than one optimum solution

- Non-Separability
  - The optimum of one of the objectives is not the optimum for the other objective/s.

- High Dimensions
  - Number of dimensions of the search space is large
Which optimization algorithms can we use?

Mathematical optimization

- Based on the property of objective function and constraint function:
  - linear programming
  - non-linear programing
- Assumes properties like differentiability etc.

Grid Search

- Divide dimensions into bins
- Choose one from each bin
- Slow and can miss important optimization opportunities
Which optimization algorithms can we use?

Evolutionary Algorithms

1. Initial Population (*Parent*)
2. While stoppingCriteria is True:
   a. Offspring = Reproduction (*Recombine* + *Mutate*)
   b. Evaluate Fitness (*Evaluate*)
   c. Replace least-fit population with new offspring (*Select*)
3. Return (Population)
Biased towards EA

- Simple implementation
  - Basic EA application can be coded up in 50 lines of python
- Distributed computation
  - Algorithms can be parallelized
- Generation of new ideas that have not been explored before

EA is most explored technique in SBSE [Harman’12]
EA is really slow!

EAs require a high number of objective function evaluations

- Evaluation of single instance of software /hardware co-design problem can take weeks [Zuluaga’13]
- Test suite generation using EA can take weeks [Harman’12]
- Popular EA (NSGA-II) taking 7 days of execution time for Aviation Models [Krall’15]
Surrogate models might be the answer?

- **Surrogates**

- **Motivation**
  - Replacement of expensive function, evaluated many times
  - Widely used in Airfoil design, CFD, reservoir planning etc.
  - No known usage in Software Engineering
Surrogate can also be used to inform

- Initialization
  - Use only the best candidates evaluated using a surrogate [Rasheed'00]
Surrogate can also be used to inform

- Initialization
  - Use only the best candidates evaluated using a surrogate [Rasheed’00]

- Recombination + Mutation
  - Create multiple children and use the fittest of them all [Loshchilov’10]
  - Create local surrogate and search locally [Abboud’01]
Surrogate can also be used to inform

- **Initialization**
  - Use only the best candidates evaluated using a surrogate [Rasheed00]

- **Recombination + Mutation**
  - Create multiple children and use the fittest of them all [Loshchilov’10]
  - Create local surrogate and search locally [Abboud’01]

- **Evaluate**
  - Multiple Surrogates [Zhou’07]
  - **WHAT** is an evaluate surrogate
To Summarize

Configuration Space → EA → Result
To Summarize

Configuration Space

Sampling → Surrogate

Surrogate → EA

EA → Result
To Summarize

Configuration Space

Sampling → Surrogate

WHAT

EA → Result
APPROACH
WHAT = Clustering + Sampling

● Phase 1: Clustering
  ○ WHERE

● Phase 2: Sampling
  ○ Random Sampling - Select any point at random
  ○ East West Sampling - Find extreme points on the dimension of highest variance
  ○ Exemplar - The point with minimum performance measure

● Phase 3: Generate Surrogate - CART
  ○ Samples selected by our sampler is used to train a CART model
Phase 1: Clustering
Phase 2: Random
Phase 3: CART
Training → Clustering
Clustering → Random
Random → CART
Testing → Phase 1

WHAT
WHAT

- Training
- Testing
- Clustering
- East West

Phase 1
Phase 2
Phase 3
Phase 1

WHAT

Training

Clustering

Exemplar

Phase 2

CART

Phase 3

Testing
WHAT

Phase 1

Training

Testing

Clustering

Phase 2

Random

East West

Exemplar

Phase 3

CART
Definition

- **Real System**
  - Features can be either True or False
  - Configuration is a set of features
  - Each configuration has a corresponding response time or **performance measure**

Configuration

- F1 = True
- F2 = False
- F3 = True

Software System

Request

Response Time = 2100 ms
Definition

- **Real System**
  - Features can be either True or False
  - Configuration is a set of features
  - Each configuration has a corresponding response time or performance measure

- **Surrogate System**
  - Configuration = independent variable
  - Performance measure = dependent variable
Phase 1: Clustering

- **Clustering via WHERE**
  - Novel near-linear time spectral learner
  - Exploits underlying lower dimensionality of search space

- **In brief:**
  - Find a dimension “d” with most variance
  - Project points to “d”
  - Split data at median “d”
  - Recurse
  - Stop when |n| < sqrt(N)

- **Future work:**
  - Fast Spectral clustering [Yan’09]
  - **In brief:**
    - Polynomial time operations
      - An initial k-means pass
      - O(N^2) operations on the centroids founds by K-means
      - Final pass: map all points to the centroids found in b
- Number of samples \( N \) = 64

Algorithm:
- Find a dimension “d” with most variance
- Project points to “d”
- Split data at median “d”
- Recurse
- Stop when \(|n| < \sqrt{N}\)
- Number of samples (N) = 64

Algorithm:

1. **Find a dimension “d” with most variance**
   - Choose point at random (initial)
   - Find furthest point (east)
   - Find furthest point from east (west)

2. Project points to “d”
3. Split data at median “d”
4. Recurse
5. Stop when |n| < sqrt(N)
- Number of samples (N) = 64

Algorithm:

- **Find a dimension “d” with most variance**
  - Choose point at random (initial)
  - Find furthest point (east)
  - Find furthest point from east (west)
- Project points to “d”
- Split data at median “d”
- Recurse
- Stop when |n| < sqrt(N)
- Number of samples (N) = 64

Algorithm:

- Find a dimension “d” with most variance
  - Choose point at random (initial)
  - Find furthest point (east)
  - Find furthest point from east (west)
- Project points to “d”
- Split data at median “d”
- Recurse
- Stop when |n| < sqrt(N)
- Number of samples (N) = 64

Algorithm:

- **Find a dimension “d” with most variance**
  - Choose point at random (initial)
  - Find furthest point (east)
  - Find furthest point from east (west)
- Project points to “d”
- Split data at median “d”
- Recurse
- Stop when |n| < sqrt(N)
- Number of samples = 64

Algorithm:

- **Find a dimension “d” with most variance**
  - Choose point at random (initial)
  - Find furthest point (east)
  - Find furthest point from east (west)
- Project points to “d”
- Split data at median “d”
- Recurse
- Stop when |n| < sqrt(N)
- Number of samples (N) = 64

Algorithm:

• Find a dimension “d” with most variance
  • Choose point at random (initial)
  • Find furthest point (east)
  • Find furthest point from east (west)

• **Project points to “d”**
  • For all points
    • Choose a point (candidate)
    • Calculate position on d dimension

• Split data at median “d”
• Recurse
• Stop when |n| < sqrt(N)
- Number of samples (N) = 64

Algorithm:

- Find a dimension “d” with most variance
  - Choose point at random (initial)
  - Find furthest point (east)
  - Find furthest point from east (west)
- **Project points to “d”**
  - For all points
    - Choose a point (candidate)
    - Calculate position on d dimension
- Split data at median “d”
- Recurse
- Stop when |n| < sqrt(N)
Number of samples (N) = 64

Algorithm:
- Find a dimension “d” with most variance
  - Choose point at random (initial)
  - Find furthest point (east)
  - Find furthest point from east (west)
- **Project points to “d”**
  - For all points
    - Choose a point (candidate)
    - Calculate position on d dimension
- Split data at median “d”
- Recurse
- Stop when |n| < sqrt(N)
- Number of samples (N) = 64

Algorithm:

- Find a dimension “d” with most variance
  - Choose point at random (initial)
  - Find furthest point (east)
  - Find furthest point from east (west)

- Project points to “d”
  - For all points
    - Choose a point (candidate)
    - Calculate position on d dimension

- Split data at median “d”
- Recurse
- Stop when |n| < sqrt(N)

\[ x = \frac{a^2 + c^2 - b^2}{2c} \]
- Number of samples (N) = 64

Algorithm:

- Find a dimension “d” with most variance
  - Choose point at random (initial)
  - Find furthest point (east)
  - Find furthest point from east (west)
- Project points to “d”
  - For all points
    - Choose a point (candidate)
    - Calculate position on d dimension
- Split data at median of “d”
- Recurse
- Stop when |n| < sqrt(N)
- Number of samples (N) = 64

Algorithm:

- Find a dimension “d” with most variance
  - Choose point at random (initial)
  - Find furthest point (east)
  - Find furthest point from east (west)
- Project points to “d”
  - For all points
    - Choose a point (candidate)
    - Calculate position on d dimension
- Split data at median of “d”
- Recurse
- Stop when |n| < sqrt(N)
- Number of samples (N) = 64

Algorithm:
- Find a dimension “d” with most variance
  - Choose point at random (initial)
  - Find furthest point (east)
  - Find furthest point from east (west)
- Project points to “d”
  - For all points
    - Choose a point (candidate)
    - Calculate position on d dimension
- Split data at median of “d”
- **Recurse**
- Stop when |n| < sqrt(N)
- Number of samples (N) = 64

Algorithm:
• Find a dimension “d” with most variance
  • Choose point at random (initial)
  • Find furthest point (east)
  • Find furthest point from east (west)
• Project points to “d”
  • For all points
    • Choose a point (candidate)
    • Calculate position on d dimension
• Split data at median of “d”
• Recurse
• Stop when |n| < sqrt(N)
Phase 2: Sampling

Choosing representative candidates from clusters

- **Random**
  - Choose a candidate at random
  - Number of evaluations/Cluster = 1
  - Point selected/Cluster = 1

- **East-West**
  - Choose extreme points in dimension of maximum variance
  - Number of evaluations/Cluster = 2
  - Point selected/Cluster = 2

- **Exemplar**
  - Choose the best candidate from the cluster
  - Number of evaluations/Cluster = n
  - Point selected/Cluster = 1
Phase 3: Generate Surrogate

- Use the configuration/s sampled from each cluster
- Run the configuration
  - In this work, we performed a table lookup
- Train a CART decision tree learner using:
  - Configurations (Independent Variable)
  - Performance Measure (Dependent Variable)
Experiments
Experiments

- **Datasets Used:**
  - Apache - *open-source Web server*
  - Berkeley DB C (*BDBC*) - *embedded database system written in C*
  - Berkeley DB Java (*BDBJ*) - *BDBC in Java with SQL support*
  - LLVM - *a compiler infrastructure written in C++*
  - SQLite - *embedded database system*
  - X264 - *is a video encoder in C*

- **Surrogate Used:** CART

- **Techniques compared against:**
  - Siegmund et al.
  - Guo et al.
  - Sarkar et al.

- **Performance Measure:**
  - MRE: Mean Relative Error
    \[ MRE = \frac{|\text{actual} - \text{predicted}|}{\text{actual}} \times 100 \]
Techniques compared against

- 2012: [Siegmund’12]
- 2013: [Guo’13]
- 2015: [Sarkar’15]
Uses Feature Wise heuristics:

- Find
  - a pair of configuration $(C_1$ and $C_2$)
  - has same features except for one ($F_i$)
- Performance score (PS) of $F_i$
  $PS(F_i) = PS(C_1) - PS(C_2)$
- Performance of a new $C_i$
  $PS(C_i) = \sum PS(F_i) \forall F_i \in C_i$
Techniques compared against

Progressive Sampling Approach:

While terminationCriteria() is True:
  ● Random Sampling
  ● Samples in step of |F|
  ● Build a CART tree
Techniques compared against

- 2012 [Sieg mund’12]
- 2013 [Guo’13]
- 2015 [Sarkar’15]

Uses Feature Frequencies:
- Projective sampling to decide number of configurations to sample
- Random Sampling
- Build a CART tree
Research Questions

RQ 1: Can WHAT generate good predictions using only a small number of configurations?

RQ 2: Do less data cause larger variances in predicted values?

RQ 3: Can “good” surrogate models (to be used in optimizers) be built using WHAT?

RQ 4: How good is WHAT compared to the state of the art predictors?
RQ1 + RQ2

RQ1 + RQ2 explore
• if WHAT can generate good predictors with low variance
• how much of data should WHAT reflect upon

Comparison between:
• Baseline (using all the data)
• WHERE + Random
• WHERE + EAST-West
• WHERE + Exemplar
Design of Experiment
Design of Experiment

Data (100)
Design of Experiment

Data (100)

Train (10)  Test (90)
Design of Experiment

Data (100)

Train (10)  Test (90)

Baseline

Table Lookup  CART  OUTPUT

WHERE

Random
Design of Experiment

Data (100)

Train (10)  Test (90)

Baseline

Table Lookup  CART

OUTPUT

WHERE

East-West
Design of Experiment

Data (100)

Train (10) — Test (90)

Baseline

Table Lookup — CART — OUTPUT

WHERE

Exemplar
Design of Experiment

Data (100)

Train (10)  Test (90)

Baseline

Table Lookup

CART

OUTPUT

WHERE

Random

East-West

Exemplar
Design of Experiment

Data (100)

Train (10) -> Baseline

Table Lookup

CART -> OUTPUT

Train (20) -> Test (80)

WHERE
- Random
- East-West
- Exemplar

Train (90) -> Test (10)
RQ1: Can WHAT generate good predictions using only a small number of configurations?
RQ1: Can WHAT generate good predictions using only a small number of configurations?
RQ1: Can WHAT generate good predictions using only a small number of configurations?

\[
\text{MRE} = \left(\frac{|\text{actual} - \text{predicted}|}{\text{actual}}\right) \times 100
\]
RQ1: Can WHAT generate good predictions using only a small number of configurations?

MRE = \frac{|\text{actual} - \text{predicted}|}{\text{actual}} \times 100
RQ1: Can WHAT generate good predictions using only a small number of configurations?

\[
MRE = \left| \frac{\text{actual} - \text{predicted}}{\text{actual}} \right| \times 100
\]
RQ1: Can WHAT generate good predictions using only a small number of configurations?

\[
\text{MRE} = \frac{|\text{actual} - \text{predicted}|}{\text{actual}} \times 100
\]
RQ1: Can WHAT generate good predictions using only a small number of configurations?

\[
\text{MRE} = \frac{|\text{actual} - \text{predicted}|}{\text{actual}} \times 100
\]
RQ1: Can WHAT generate good predictions using only a small number of configurations?

MRE = \frac{|\text{actual} - \text{predicted}|}{\text{actual}} \times 100
RQ1: Can WHAT generate good predictions using only a small number of configurations?
RQ1: Can WHAT generate good predictions using only a small number of configurations?
RQ1: Can WHAT generate good predictions using only a small number of configurations?
RQ2: Do less data cause larger variances in predicted values?
RQ2: Do less data cause larger variances in predicted values?
RQ2: Do less data cause larger variances in predicted values?

![Graph showing standard deviation with different percentage of data reflected upon for Software Systems S1, S2, S3, S4, and Baseline.](image)
RQ2: Do less data cause larger variances in predicted values?

![Graph showing standard deviation with percentage of data reflected upon.](image)
RQ2: Do less data cause larger variances in predicted values?

![Graph showing standard deviation with different datasets and percentage of data reflected upon. The graph compares various software systems such as Apache, BDDB, LLVM, SQL, X86, and a baseline. The y-axis represents standard deviation, and the x-axis represents the percentage of data.]
RQ2: Do less data cause larger variances in predicted values?

Software Systems

Standard Deviation

% of data reflected upon
RQ2: Do less data cause larger variances in predicted values?
RQ2: Do less data cause larger variances in predicted values?
**RQ2:** Do less data cause larger variances in predicted values?
RQ2: Do less data cause larger variances in predicted values?
RQ2: Do less data cause larger variances in predicted values?

**Random**

<table>
<thead>
<tr>
<th>Software System</th>
<th>Apache</th>
<th>BDBC</th>
<th>BDBJ</th>
<th>LLVM</th>
<th>SQLite</th>
<th>X264</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mean MRE</td>
<td>✔</td>
<td>✗</td>
<td>✔</td>
<td>✔</td>
<td>✗</td>
<td>✔</td>
</tr>
<tr>
<td>Standard Deviation</td>
<td>?</td>
<td>✗</td>
<td>?</td>
<td>?</td>
<td>✔</td>
<td>?</td>
</tr>
</tbody>
</table>

**East-West**

<table>
<thead>
<tr>
<th>Software System</th>
<th>Apache</th>
<th>BDBC</th>
<th>BDBJ</th>
<th>LLVM</th>
<th>SQLite</th>
<th>X264</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mean MRE</td>
<td>✗</td>
<td>✔</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>✔</td>
</tr>
<tr>
<td>Standard Deviation</td>
<td>?</td>
<td>✔</td>
<td>?</td>
<td>?</td>
<td>✔</td>
<td>?</td>
</tr>
</tbody>
</table>

**Exemplar**

<table>
<thead>
<tr>
<th>Software System</th>
<th>Apache</th>
<th>BDBC</th>
<th>BDBJ</th>
<th>LLVM</th>
<th>SQLite</th>
<th>X264</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mean MRE</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
</tr>
<tr>
<td>Standard Deviation</td>
<td>?</td>
<td>✔</td>
<td>?</td>
<td>?</td>
<td>✔</td>
<td>?</td>
</tr>
</tbody>
</table>
RQ2: Do less data cause larger variances in predicted values?

![Graph showing variances in predicted values for different systems and data sets.](image)

<table>
<thead>
<tr>
<th>Software System</th>
<th>Apache</th>
<th>BDGC</th>
<th>BDJ</th>
<th>LLVM</th>
<th>SQLite</th>
<th>X264</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Mean MRE</strong></td>
<td>✔️</td>
<td>✗</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✗</td>
</tr>
<tr>
<td><strong>Standard Deviation</strong></td>
<td>✔️</td>
<td>✗</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Software System</th>
<th>Apache</th>
<th>BDGC</th>
<th>BDJ</th>
<th>LLVM</th>
<th>SQLite</th>
<th>X264</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>East-West</strong></td>
<td>✗</td>
<td>✔️</td>
<td>✗</td>
<td>✗</td>
<td>✔️</td>
<td>✔️</td>
</tr>
<tr>
<td><strong>Mean MRE</strong></td>
<td>✗</td>
<td>✔️</td>
<td>✗</td>
<td>✗</td>
<td>✔️</td>
<td>✔️</td>
</tr>
<tr>
<td><strong>Standard Deviation</strong></td>
<td>✗</td>
<td>✔️</td>
<td>✗</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Software System</th>
<th>Apache</th>
<th>BDGC</th>
<th>BDJ</th>
<th>LLVM</th>
<th>SQLite</th>
<th>X264</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Exemplar</strong></td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>✔️</td>
<td>✔️</td>
</tr>
<tr>
<td><strong>Mean MRE</strong></td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>✔️</td>
<td>✔️</td>
</tr>
<tr>
<td><strong>Standard Deviation</strong></td>
<td>✗</td>
<td>✔️</td>
<td>✗</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
</tr>
</tbody>
</table>
RQ1 + RQ2: Observations

- Baseline results is the best
  - It uses 100% of data
- Results plateaued after 40%
- WHERE + Exemplar
  - largest Mean MRE
  - Not Recommended
- WHERE + East-West
  - MRE 3/6 times better/similar
  - Standard deviation is low
  - Recommended
- WHERE + Random
  - MRE 4/6 times better/similar
  - Standard deviation is low
  - Recommended
RQ1 + RQ2: Evaluation

- WHERE + East-West
  - MRE 3/6 times better/similar
  - Standard deviation is low
  - Recommended

- WHERE + Random
  - MRE 4/6 times better/similar
  - Standard deviation is low
  - Recommended
RQ1 + RQ2: Evaluation

- **WHERE + East-West**
  - MRE 3/6 times better/similar
  - Standard deviation is low
  - Recommended

- **WHERE + Random**
  - MRE 4/6 times better/similar
  - Standard deviation is low
  - Recommended

(When Training Data = 40%)
RQ1 + RQ2: Evaluation

- WHERE + East-West
  - MRE 3/6 times better/similar
  - Standard deviation is low
  - **Recommended**

- WHERE + Random
  - MRE 4/6 times better/similar
  - Standard deviation is low
  - **Recommended**

(When Training Data = 40%)
RQ1 + RQ2: Evaluation

- WHERE + East-West
  - MRE 3/6 times better/similar
  - Standard deviation is low
  - Recommended

- WHERE + Random
  - MRE 4/6 times better/similar
  - Standard deviation is low
  - Recommended

Exemplar uses one point/cluster but to find the exemplar point you need to evaluate all.

Random Sampling evaluates half the configuration compared to EastWest Sampling.

Software Systems
RQ 3: Can “good” surrogate models (to be used in optimizers) be built using WHAT?

RQ 3 explore
- if predictors generated using samples from WHAT can find faster performance scores (eg. Response time)

Optimization Goal
- Minimize the performance score of the system

Comparison between:
- GALE [Krall’15]
- DE [Storn’95]
- NSGA-II [Deb’02]
RQ 3: Can “good” surrogate models (to be used in optimizers) be built using WHAT?

Instances sorted based on Performance Scores
RQ 3: Can “good” surrogate models (to be used in optimizers) be built using WHAT?

Instances sorted based on Performance Scores
RQ 3: Can “good” surrogate models (to be used in optimizers) be built using WHAT?

Configuration found by optimizers projected onto ground truth

Performance Scores

Instances sorted based on Performance Scores
RQ 3: Can “good” surrogate models (to be used in optimizers) be built using WHAT?

- Optimization Goal: Minimization
RQ 3: Can “good” surrogate models (to be used in optimizers) be built using WHAT?

- Optimization Goal: Minimization

- Optimized configurations
  - within 1% of the fastest configuration
RQ 4: How good is WHAT compared to the state of the art predictors?

RQ 4 explores

- If WHAT is better than state-of-the-art techniques
  - Siegmund et al. - FW heuristics
  - Guo et al. - Progressive Sampling
  - Sarkar et al. - Random Sampling + Feature-wise heuristics
RQ 4: How good is WHAT compared to the state of the art predictors?

RQ 4 explores

- If WHAT is better than state-of-the-art techniques
  - Siegmund et al. - FW heuristics
  - Guo et al. - Progressive Sampling
  - Sarkar et al. - Random Sampling + Feature-wise heuristics

Different stopping criteria

- **PW**: Pair Wise
- **2N**: 2 x # of Features
RQ 4: How good is WHAT compared to the state of the art predictors?

RQ 4 explores
- If WHAT is better than state-of-the-art techniques
  - Siegmund et al. - FW heuristics
  - Guo et al. - Progressive Sampling
  - Sarkar et al. - Random Sampling + Feature-wise heuristics
RQ 4: How good is WHAT compared to the state of the art predictors?

RQ 4 explores

- If WHAT is better than state-of-the-art techniques
  - Siegmund et al. - FW heuristics
  - Guo et al. - Progressive Sampling
  - Sarkar et al. - Random Sampling + Feature-wise heuristics
RQ 4: How good is WHAT compared to the state of the art predictors?

RQ 4 explores

- If WHAT is better than state-of-the-art techniques
  - Siegmund et al. - FW heuristics
  - Guo et al. - Progressive Sampling
  - Sarkar et al. - Random Sampling + Feature-wise heuristics
RQ 4: How good is WHAT compared to the state of the art predictors?
RQ 4: How good is WHAT compared to the state of the art predictors?

![Graph showing Mean(%), Standard Deviation, and Measurement (%) with Configuration for various software systems over three years (2012, 2013, 2015). The graphs compare different predictors (Siegmund, Guo, Sarkar, WHAT) across Apache, BDB, BDBJ, LLVM, SQLite, X264. The y-axis is on a log scale.](image-url)
RQ 4: How good is WHAT compared to the state of the art predictors?

Software Systems

Percentage Measure (log scale)
RQ 4: How good is WHAT compared to the state of the art predictors?
RQ 4: How good is WHAT compared to the state of the art predictors?
RQ 4: How good is WHAT compared to the state of the art predictors?
RQ 4: How good is WHAT compared to the state of the art predictors?
RQ 4: How good is WHAT compared to the state of the art predictors?

[Sieg mund’12] [Guo’13] [Sarkar’15]
RQ 4: How good is WHAT compared to the state of the art predictors?
RQ 4: How good is WHAT compared to the state of the art predictors?
RQ 4: How good is WHAT compared to the state of the art predictors?
RQ 4: How good is WHAT compared to the state of the art predictors?
RQ 4: How good is WHAT compared to the state of the art predictors?
RQ 4: How good is WHAT compared to the state of the art predictors?
RQ 4: How good is WHAT compared to the state of the art predictors?
RQ 4: How good is WHAT compared to the state of the art predictors?

2012

[Siegmund'12]

2013

[Guo'13]

2015

[Sarkar'15]
RQ 4: How good is WHAT compared to the state of the art predictors?
Research Questions

RQ 1: Can WHAT generate good predictions using only a small number of configurations?

RQ 2: Do less data cause larger variances in predicted values?

RQ 3: Can “good” surrogate models (to be used in optimizers) be built using WHAT?

RQ 4: How good is WHAT compared to the state of the art predictors?
Research Questions

RQ 1: Can WHAT generate good predictions using only a small number of configurations?  
YES

RQ 2: Do less data cause larger variances in predicted values?

RQ 3: Can “good” surrogate models (to be used in optimizers) be built using WHAT?

RQ 4: How good is WHAT compared to the state of the art predictors?
Research Questions

RQ 1: Can WHAT generate good predictions using only a small number of configurations? **YES**

RQ 2: Do less data cause larger variances in predicted values? **NO**

RQ 3: Can “good” surrogate models (to be used in optimizers) be built using WHAT?

RQ 4: How good is WHAT compared to the state of the art predictors?
Research Questions

RQ 1: Can WHAT generate good predictions using only a small number of configurations?  YES

RQ 2: Do less data cause larger variances in predicted values?  NO

RQ 3: Can “good” surrogate models (to be used in optimizers) be built using WHAT?  YES

RQ 4: How good is WHAT compared to the state of the art predictors?
Research Questions

RQ 1: Can WHAT generate good predictions using only a small number of configurations?  
YES

RQ 2: Do less data cause larger variances in predicted values?  
NO

RQ 3: Can “good” surrogate models (to be used in optimizers) be built using WHAT?  
YES

RQ 4: How good is WHAT compared to the state of the art predictors?  
Comparable
Future Work
Future Work

- **Progressive WHAT**
  - WHAT is rigid
  - Has no options of budget
  - Progressive Sampling using WHAT

- **Multi-objective Problems**
  - Problem are multi-objective
  - New surrogates required
  - New surrogate model update techniques

- **Sampling Way**
  - Sampling is preferable if evaluation is expensive
  - Initial results are competitive with other algorithms

- **Spectral Grid Search**
  - Exploit the underlying dimension while generating Grids
RQ 1: Can WHAT generate good predictions using only a small number of configurations?  
YES

RQ 2: Do less data cause larger variances in predicted values?  
NO

RQ 3: Can “good” surrogate models (to be used in optimizers) be built using WHAT?  
YES

RQ 4: How good is WHAT compared to the state of the art predictors?  
Comparable

Question and Comments
References

- [Abboud'01] Abboud, Schoenauer. Surrogate deterministic mutation: Preliminary results, Artificial Evolution '01