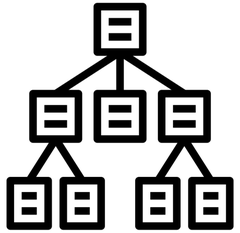
White-box Analysis for Modeling and Debugging the Performance of Configurable Systems

Thesis Proposal

Miguel Velez

Most software is configurable

Configurable System



Indexing

Encryption

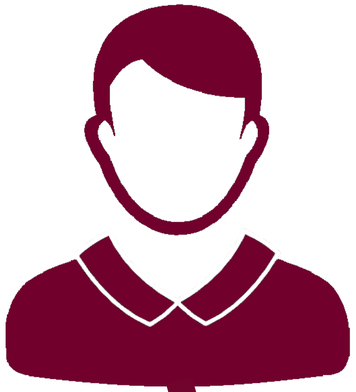
Compression

Logging

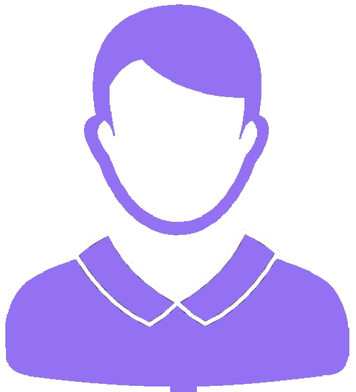
Core functionality

Design decisions affect functionality and quality attributes

Different Users Have Different Needs



Configuration:



Configuration:

Configuration:

Configurable System

Indexing

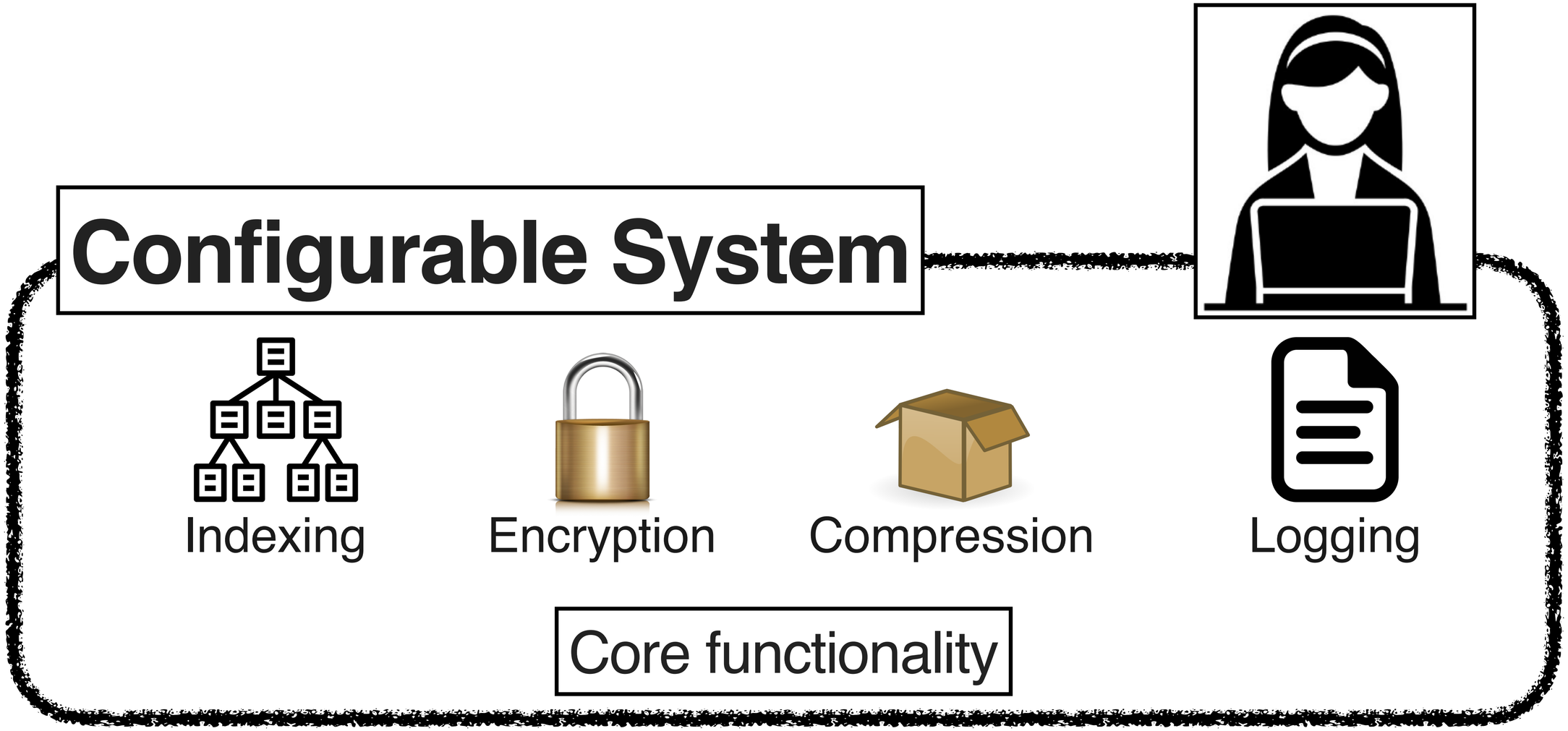
Encryption

Compression

Logging

Core functionality

Developers Implement a Single System



Developers Implement a Single System

But

Logging

Core functionality

Home Download Documentation Community Blog

Manage massive amounts of data,
fast, without losing sleep

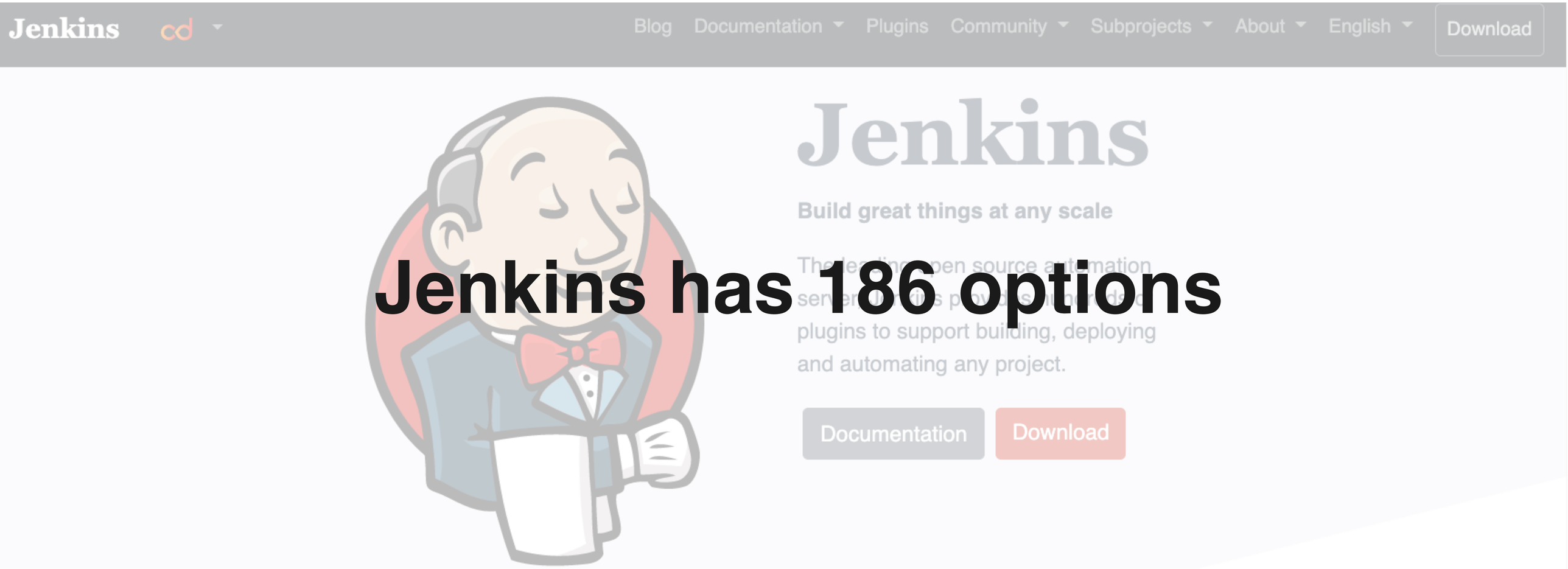
Cassandra has 170 options

Download Cassandra

Cassandra 4.0-beta3 Changelog

What is Cassandra?

The Apache Cassandra database is the right choice when you need scalability and high availability without compromising performance. Linear scalability and proven fault-tolerance on commodity hardware or cloud infrastructure make it the perfect platform for mission-critical data. Cassandra's support for replicating across multiple datacenters is best-in-class, providing lower latency for your users and the peace of mind of knowing that you can survive



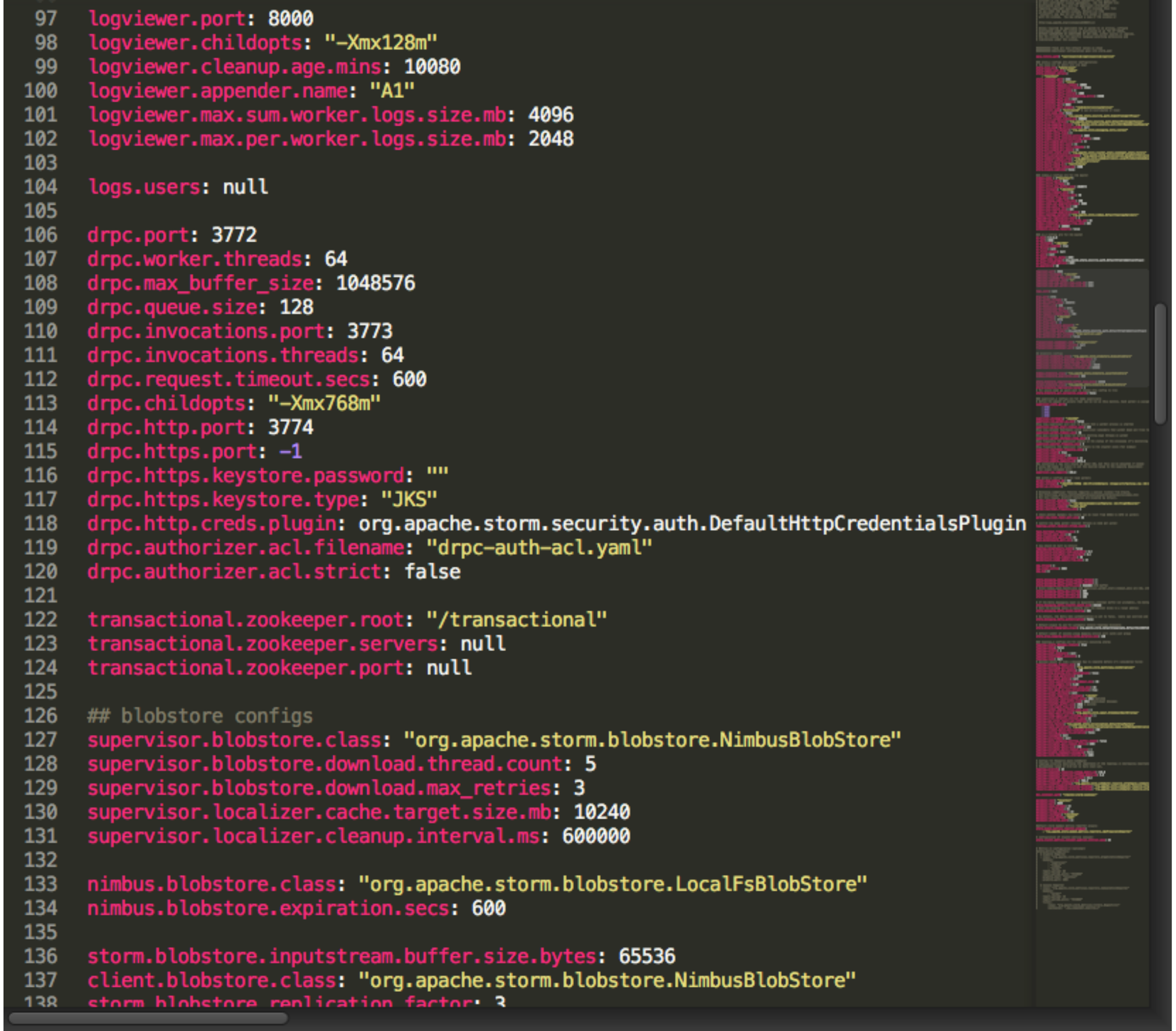
Jenkins has 186 options

APACHE KAFKA

Kafka's Broker has 208 options

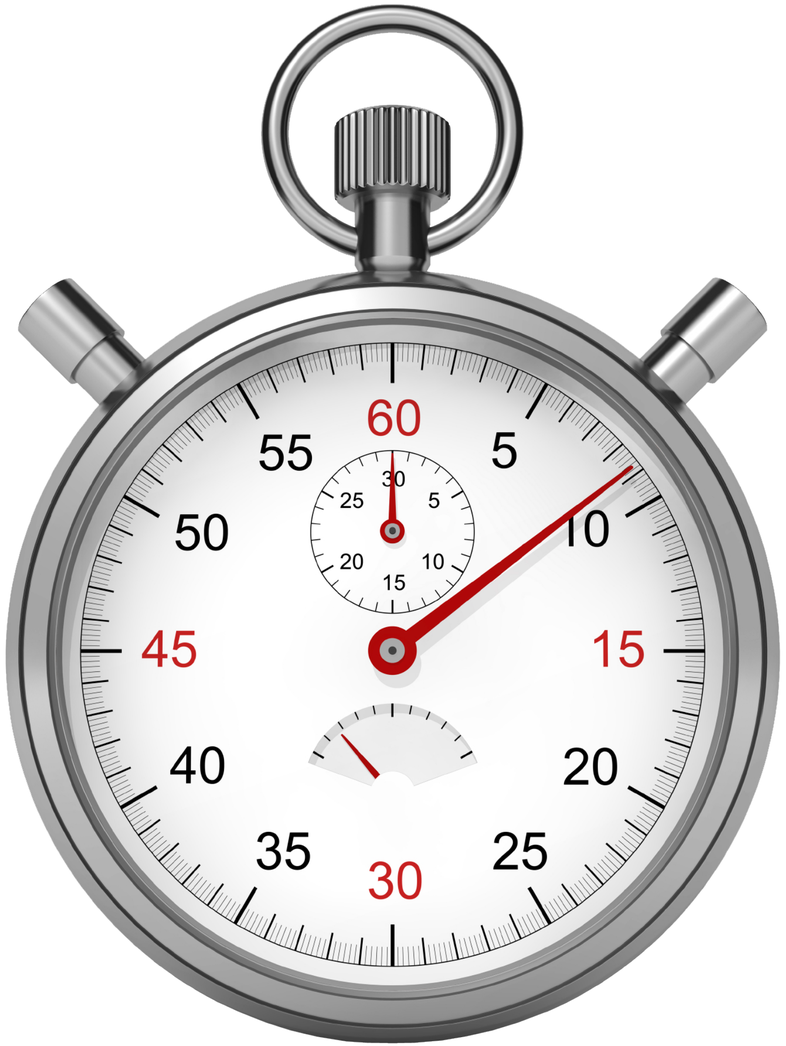
Apache Kafka is an open-source distributed event streaming platform used by thousands of companies for high-performance data pipelines, streaming analytics, data integration, and mission-critical applications.

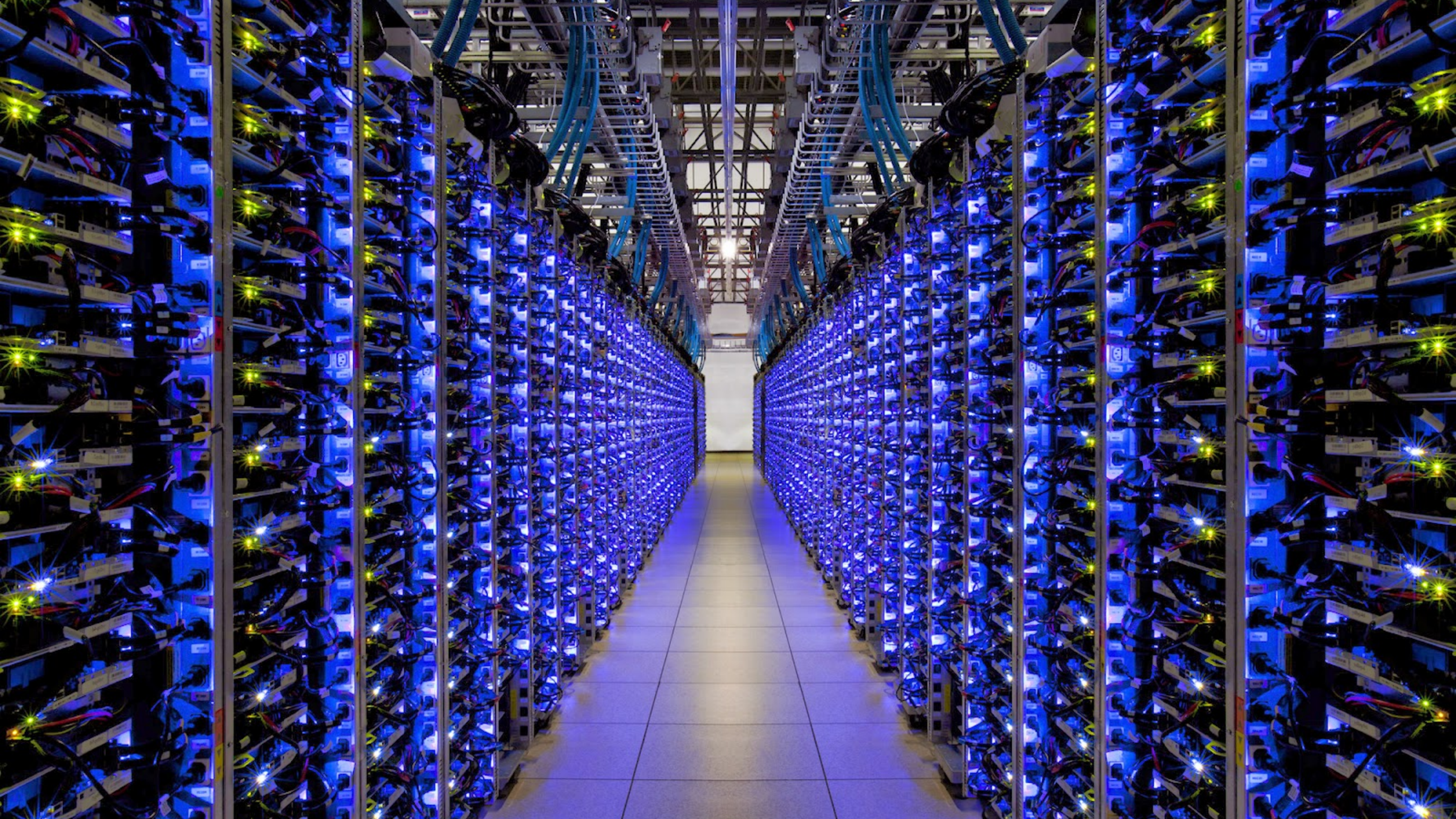


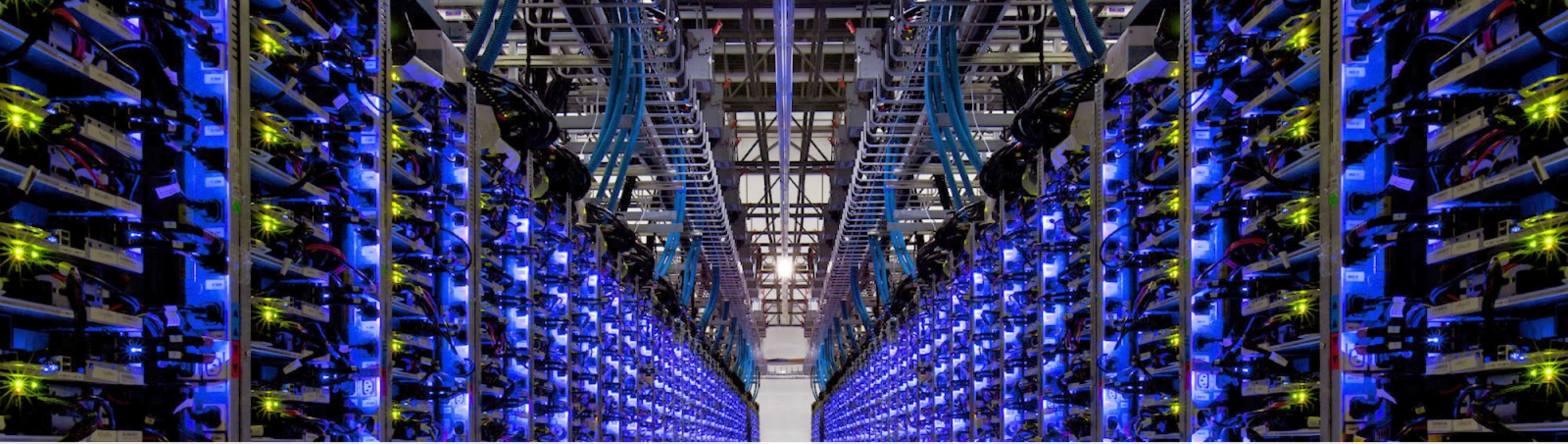



```
logviewer.port: 8000
logviewer.childopts: "-Xmx128m"
logviewer.cleanup.age.mins: 10080
logviewer.appender.name: "A1"
logviewer.max.sum.worker.logs.size.mb: 4096
logviewer.max.per.worker.logs.size.mb: 2048

logs.users: null

drpc.port: 3772
drpc.worker.threads: 64
drpc.max_buffer_size: 1048576
drpc.queue.size: 128
drpc.invocations.port: 3773
drpc.invocations.threads: 64
drpc.request.timeout.secs: 600
drpc.childopts: "-Xmx768m"
drpc.http.port: 3774
drpc.https.port: -1
drpc.https.keystore.password: ""
drpc.https.keystore.type: "JKS"
drpc.http.creds.plugin: org.apache.storm.security.auth.DefaultHttpCredentialsPlugin
drpc.authorizer.acl.filename: "drpc-auth-acl.yaml"
drpc.authorizer.acl.strict: false

transactional.zookeeper.root: "/transactional"
transactional.zookeeper.servers: null
transactional.zookeeper.port: null

## blobstore configs
supervisor.blobstore.class: "org.apache.storm.blobstore.NimbusBlobStore"
supervisor.blobstore.download.thread.count: 5
supervisor.blobstore.download.max_retries: 3
supervisor.localizer.cache.target.size.mb: 10240
supervisor.localizer.cleanup.interval.ms: 600000

nimbus.blobstore.class: "org.apache.storm.blobstore.LocalFsBlobStore"
nimbus.blobstore.expiration.secs: 600

storm.blobstore.inputstream.buffer.size.bytes: 65536
client.blobstore.class: "org.apache.storm.blobstore.NimbusBlobStore"
storm.blobstore.replication.factor: 3
```

Challenging to understand how options affect functionality and quality attributes in exponentially large configuration spaces


60
5
10
15
20
25
30
35
40
45
50
55
30
5
10
15
20
25





Performance

Execution time

Energy consumption

Operational costs

User

- Make tradeoff decisions
- Answer what-if questions
- Run system efficiently
- Satisfy their needs and requirements

Developer

- Design, implement, and maintain efficient software
- Debug surprising performance behavior

Configuration space problem

Describe performance in exponentially large configuration spaces

User

- Make good decisions
- Answer what-if questions
- Run system efficiently
- Satisfy their needs and requirements

Developer

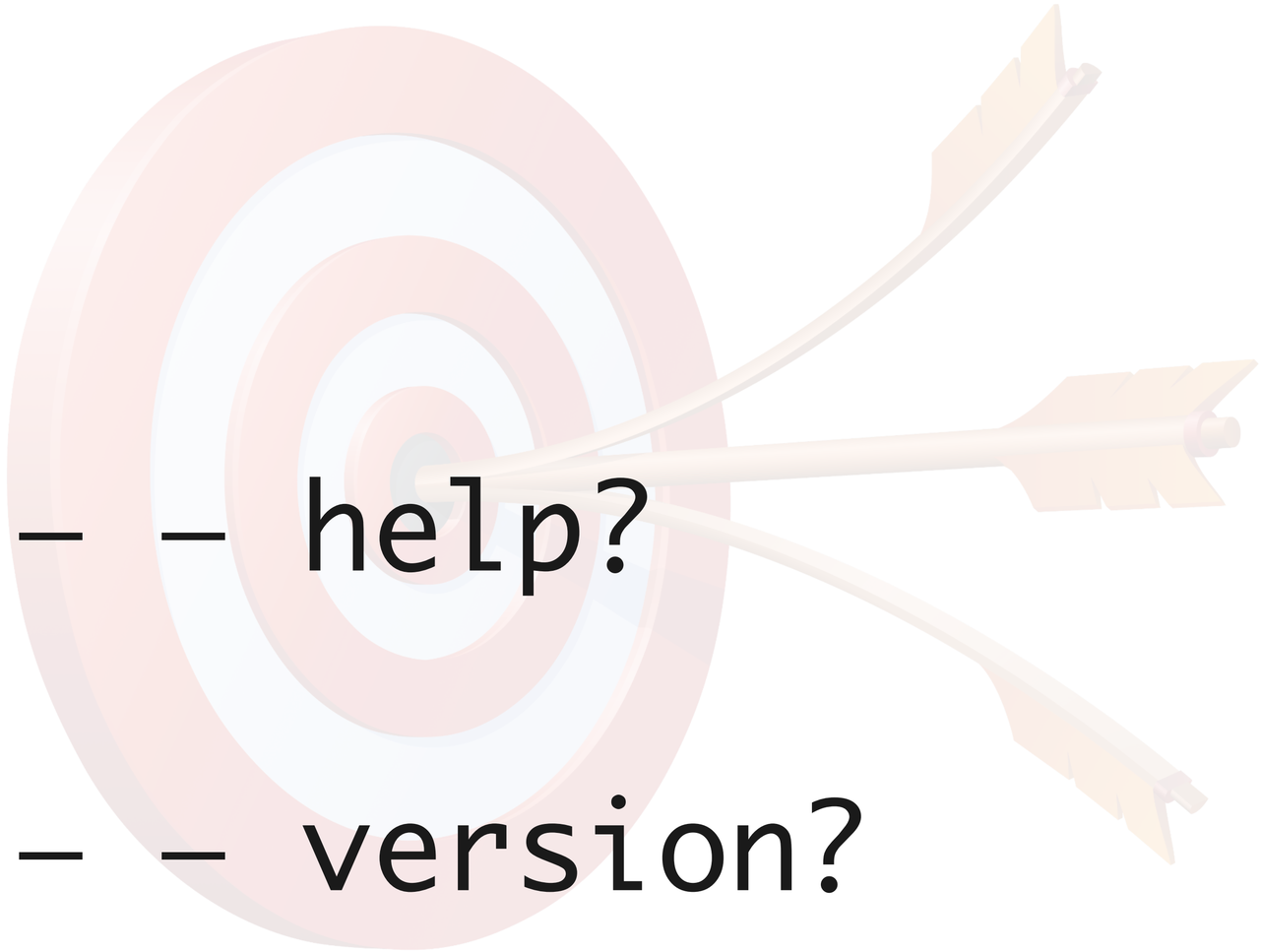
- Design, implement, and maintain efficient software
- Debug surprising behavior

Optimization!

Optimization!

Reduces:

- **Execution time**
- **Energy consumption**
- **Operational cost**



```
– –help?

– –version?
```

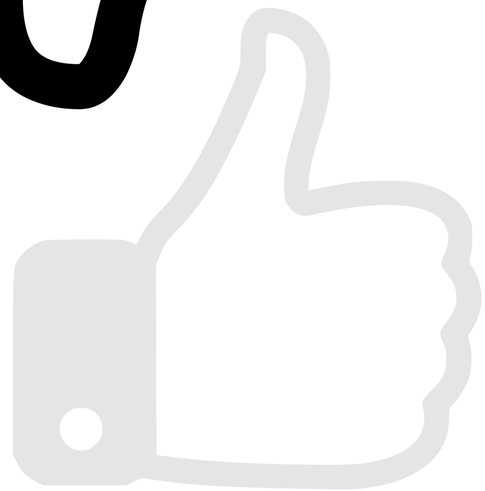
Optimization!

Reduces:

Execution time

Energy consumption

Operation cost



User

- Make informed tradeoff decisions
- Run systems efficiently

Developer

- Debug surprising behavior
- Develop and maintain efficient software

Understand how options affect performance

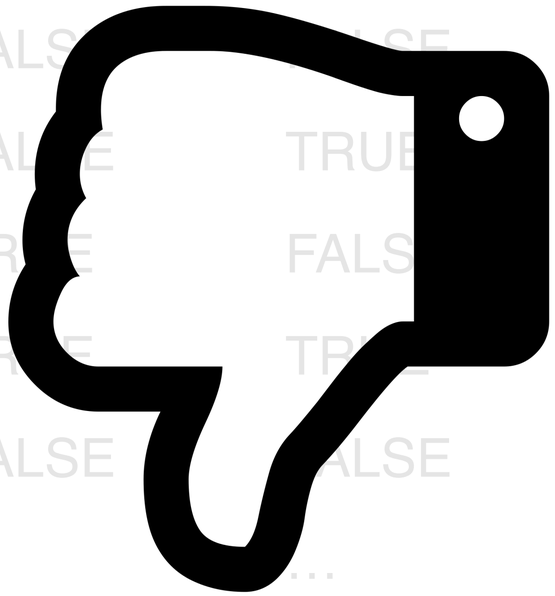
Brute-force Approach

Indexing	Encryption	Compression	Logging	…	Time
TRUE	FALSE	FALSE	FALSE	…	25
TRUE	FALSE	FALSE	TRUE	…	34
TRUE	FALSE	TRUE	FALSE	…	28
TRUE	FALSE	TRUE	TRUE	…	37
TRUE	TRUE	FALSE	FALSE	…	20
TRUE	TRUE	FALSE	TRUE	…	29
TRUE	TRUE	TRUE	FALSE	…	23
TRUE	TRUE	TRUE	TRUE	…	32
FALSE	FALSE	FALSE	FALSE	…	25
…	…	…	…	…	…

Brute-force Approach

Indexing	Encryption	Compression	Logging	…	Time
TRUE	FALSE	FALSE	FALSE	…	25
TRUE	FALSE	FALSE	TRUE	…	34
TRUE	FALSE	TRUE	FALSE	…	28
TRUE	FALSE	TRUE	TRUE	…	37
TRUE	TRUE	FALSE	FALSE	…	20
TRUE	TRUE	FALSE	TRUE	…	29
TRUE	TRUE	TRUE	FALSE	…	23
TRUE	TRUE	TRUE	TRUE	…	32
FALSE	FALSE	FALSE	FALSE	…	25
…	…	…	…	…	…

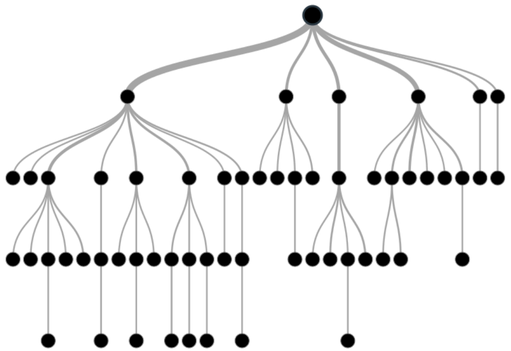
Exponentially large configuration spaces



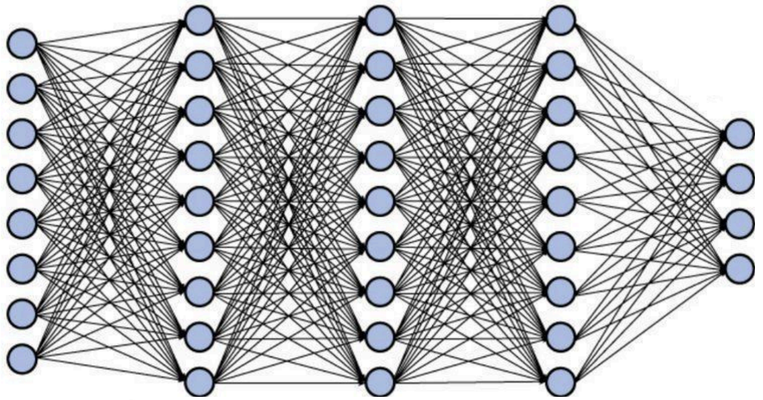
Performance-Influence Models

```
T = 25
    + 3·Logging
    - 5·Indexing
    + 9·Compression·Encryption
```

Sparse linear models



Decision trees/
Random forests



Neural networks

Performance-Influence Models

```
T = 25
+ 5·Logging
+ 3·Indexing
+ 9·Compression·Encryption
```

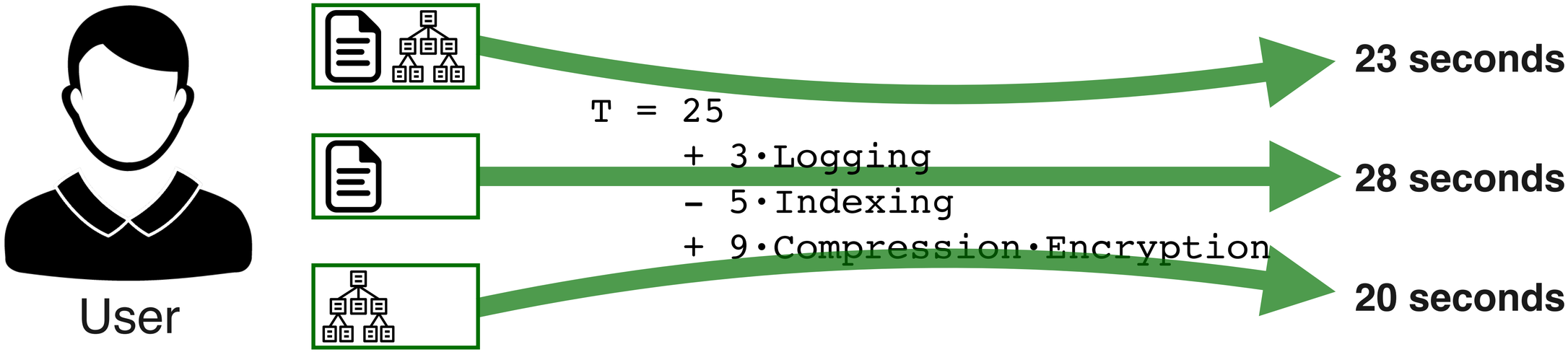
Sparse linear models

Neural networks

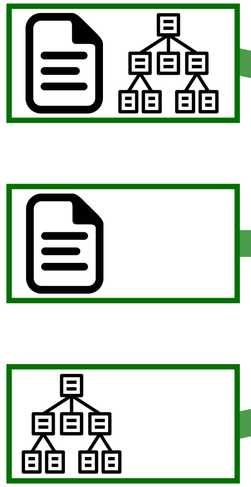
Useful for:

- **Making predictions**
- **Understanding effect of options**
- **Making tradeoff decisions**

Decision trees/
Random forests


User
T = 25
+ 3·Logging
- 5·Indexing
+ 9·Compression·Encryption
23 seconds
28 seconds
20 seconds

User



```
T = 25
    + 3·Logging
    - 5·Indexing
    + 9·Compression·Encryption
```

23 seconds

28 seconds

20 seconds

Developer

```
T = 25
    + 3·Logging
    - 5·Indexing
    + 9·Compression·Encryption
```

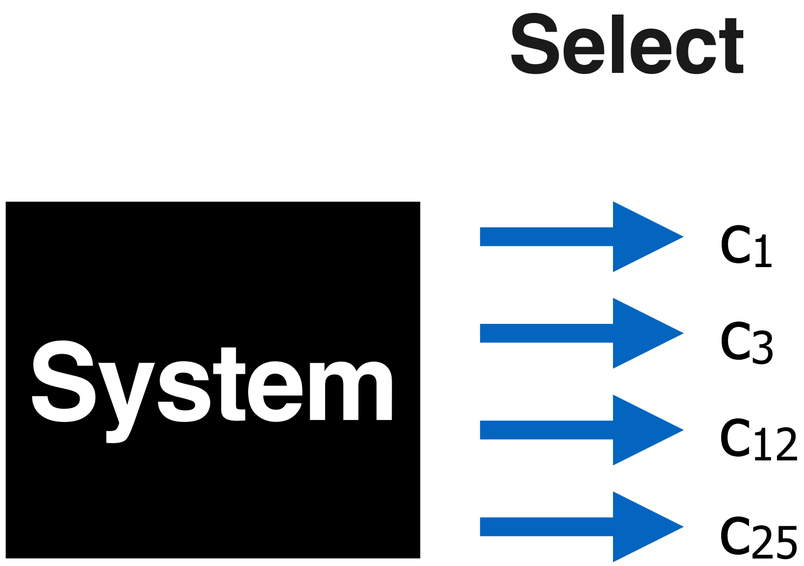
25 seconds

34 seconds

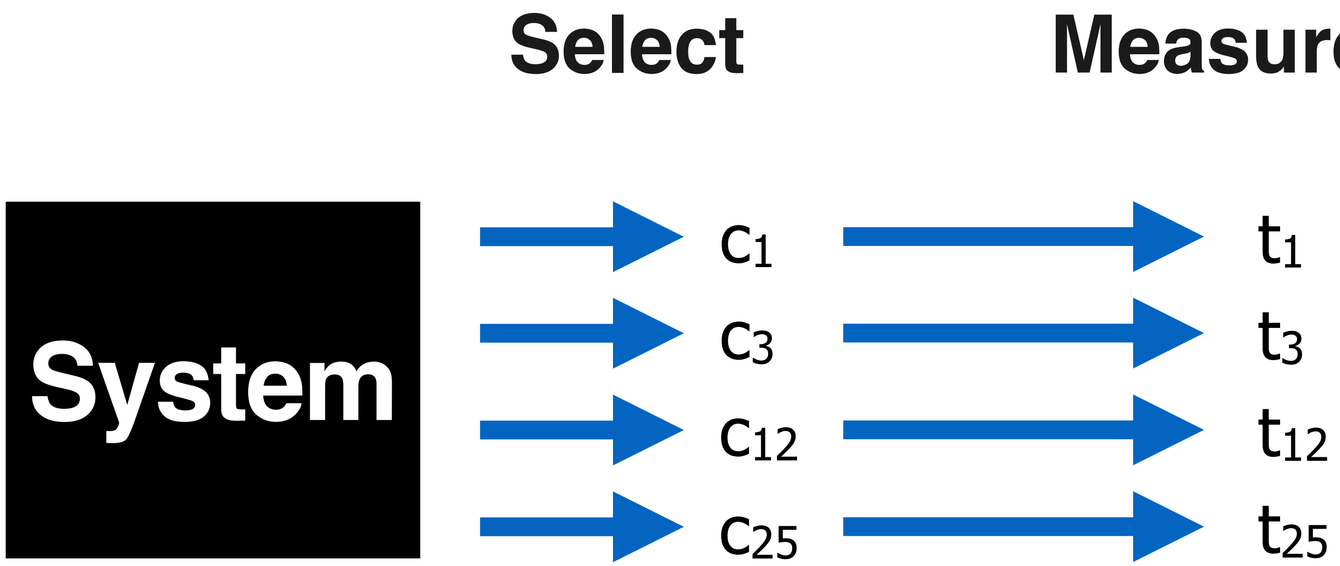
20 seconds

Black-box Approaches

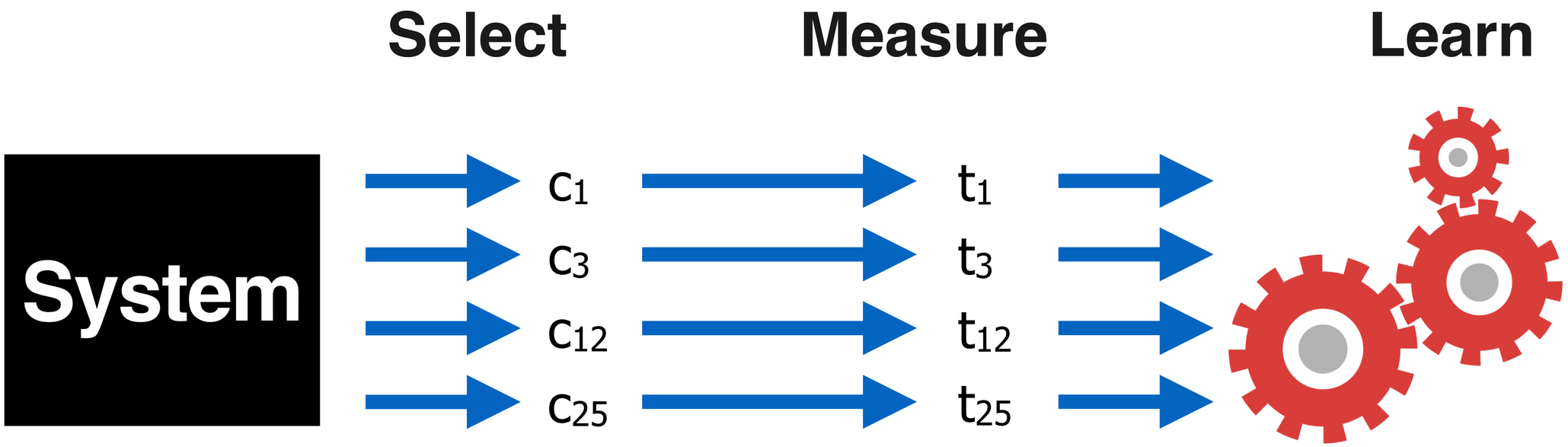
Black-box Approaches



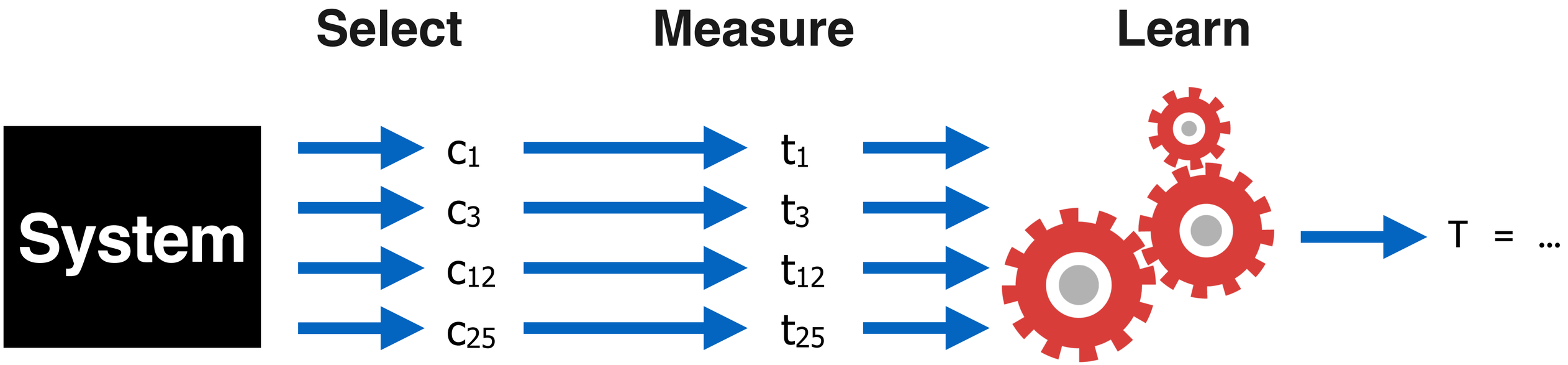
Black-box Approaches



Black-box Approaches



Black-box Approaches



```
T = 25
    + 3·Logging
    - 5·Indexing
    + 9·Compression·Encryption
```

Indexing	Encryption	Compression	Logging	…	Time
TRUE	FALSE	FALSE	FALSE	…	20
TRUE	FALSE	FALSE	TRUE	…	23
TRUE	FALSE	TRUE	FALSE	…	20
TRUE	FALSE	TRUE	TRUE	…	23
TRUE	TRUE	FALSE	FALSE	…	20
TRUE	TRUE	FALSE	TRUE	…	23
TRUE	TRUE	TRUE	FALSE	…	29
TRUE	TRUE	TRUE	TRUE	…	32
FALSE	FALSE	FALSE	FALSE	…	25
…	…	…	…	…	…

```
T = 25
    + 3·Logging
    - 5·Indexing
    + 9·Compression·Encryption
```

Indexing	Encryption	Compression	Logging	…	Time
TRUE	FALSE	FALSE	FALSE	…	20
TRUE	FALSE	FALSE	TRUE	…	23
TRUE	FALSE	TRUE	FALSE	…	20
TRUE	FALSE	TRUE	TRUE	…	23
TRUE	TRUE	FALSE	FALSE	…	20
TRUE	TRUE	FALSE	TRUE	…	23
TRUE	TRUE	TRUE	FALSE	…	29
TRUE	TRUE	TRUE	TRUE	…	32
FALSE	FALSE	FALSE	FALSE	…	25
…	…	…	…	…	…

```
T = 25
    + 3·Logging
    - 5·Indexing
    + 9·Compression·Encryption
```

Indexing	Encryption	Compression	Logging	…	Time
TRUE	FALSE	FALSE	FALSE	…	20
TRUE	FALSE	FALSE	TRUE	…	23
TRUE	FALSE	TRUE	TRUE	…	23
TRUE	TRUE	FALSE	FALSE	…	20
TRUE	TRUE	FALSE	TRUE	…	23
TRUE	TRUE	TRUE	FALSE	…	29
TRUE	TRUE	TRUE	TRUE	…	32
FALSE	FALSE	FALSE	FALSE	…	25
…	…	…	…	…	…

Might explore irrelevant interactions

```
T = 25
    + 3·Logging
    - 5·Indexing
    + 9·Compression·Encryption
```

Indexing	Encryption	Compression	Logging	…	Time
TRUE	FALSE	FALSE	FALSE	…	20
TRUE	FALSE	FALSE	TRUE	…	23
TRUE	FALSE	TRUE	FALSE	…	20
TRUE	FALSE	TRUE	TRUE	…	23
TRUE	TRUE	FALSE	FALSE	…	20
TRUE	TRUE	FALSE	TRUE	…	23
TRUE	TRUE	TRUE	FALSE	…	29
TRUE	TRUE	TRUE	TRUE	…	32
FALSE	FALSE	FALSE	FALSE	…	25
…	…	…	…	…	…

```
T = 25
    + 3·Logging
    - 5·Indexing
    + 9·Compression·Encryption
```

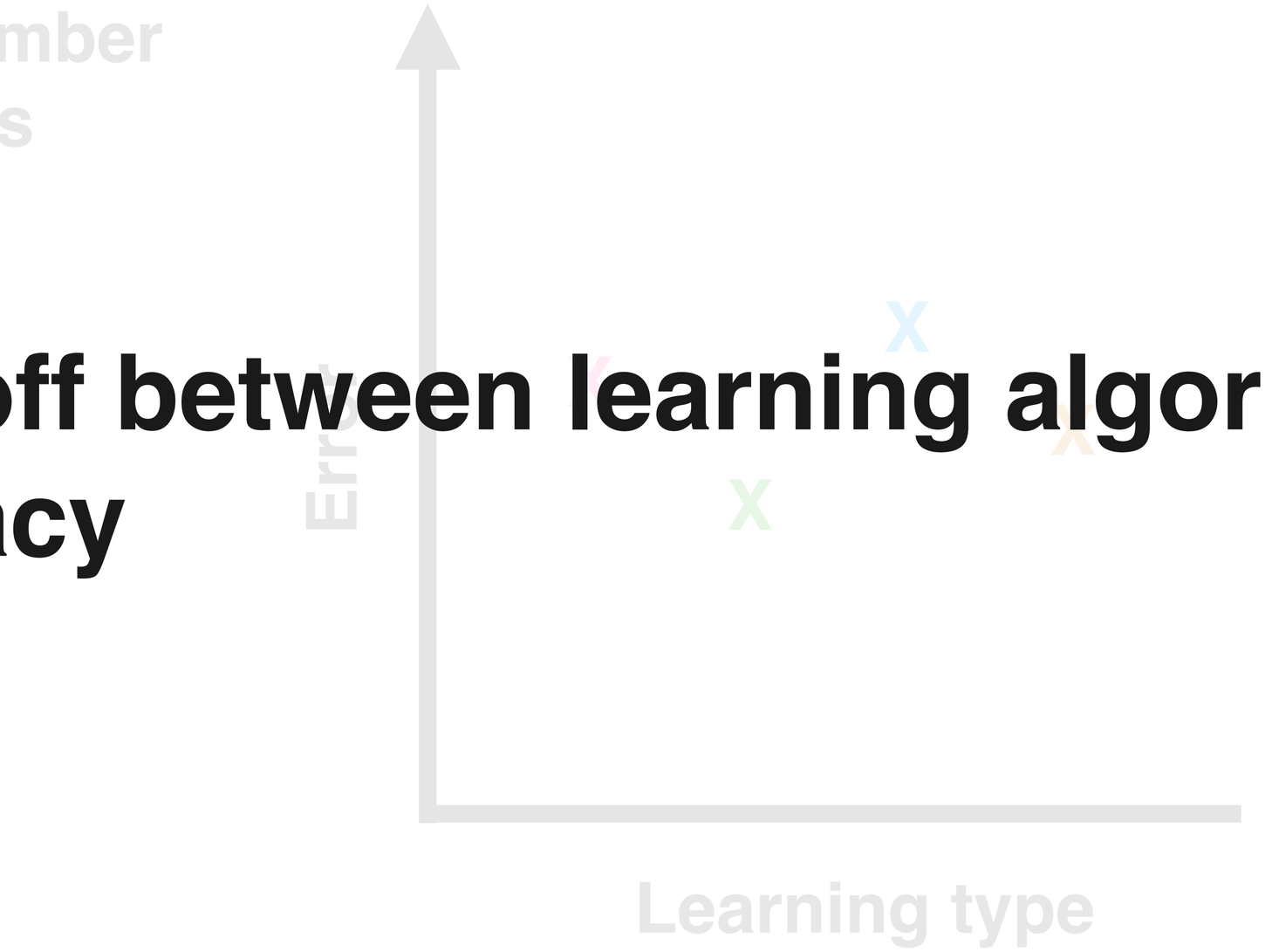
Indexing	Encryption	Compression	Logging	…	Time
TRUE	FALSE	FALSE	FALSE	…	20
TRUE	FALSE	FALSE	TRUE	…	23
TRUE	FALSE	TRUE	FALSE	…	20
TRUE	FALSE	TRUE	TRUE	…	23
TRUE	TRUE	FALSE	FALSE	…	20
TRUE	TRUE	FALSE	TRUE	…	23
TRUE	TRUE	TRUE	FALSE	…	29
TRUE	TRUE	TRUE	TRUE	…	32
FALSE	FALSE	FALSE	FALSE	…	25
…	…	…	…	…	…

Might miss relevant performance interactions

$$T = 25 + 3 \cdot Logging - 5 \cdot Indexing + 9 \cdot Compression \cdot Encryption$$

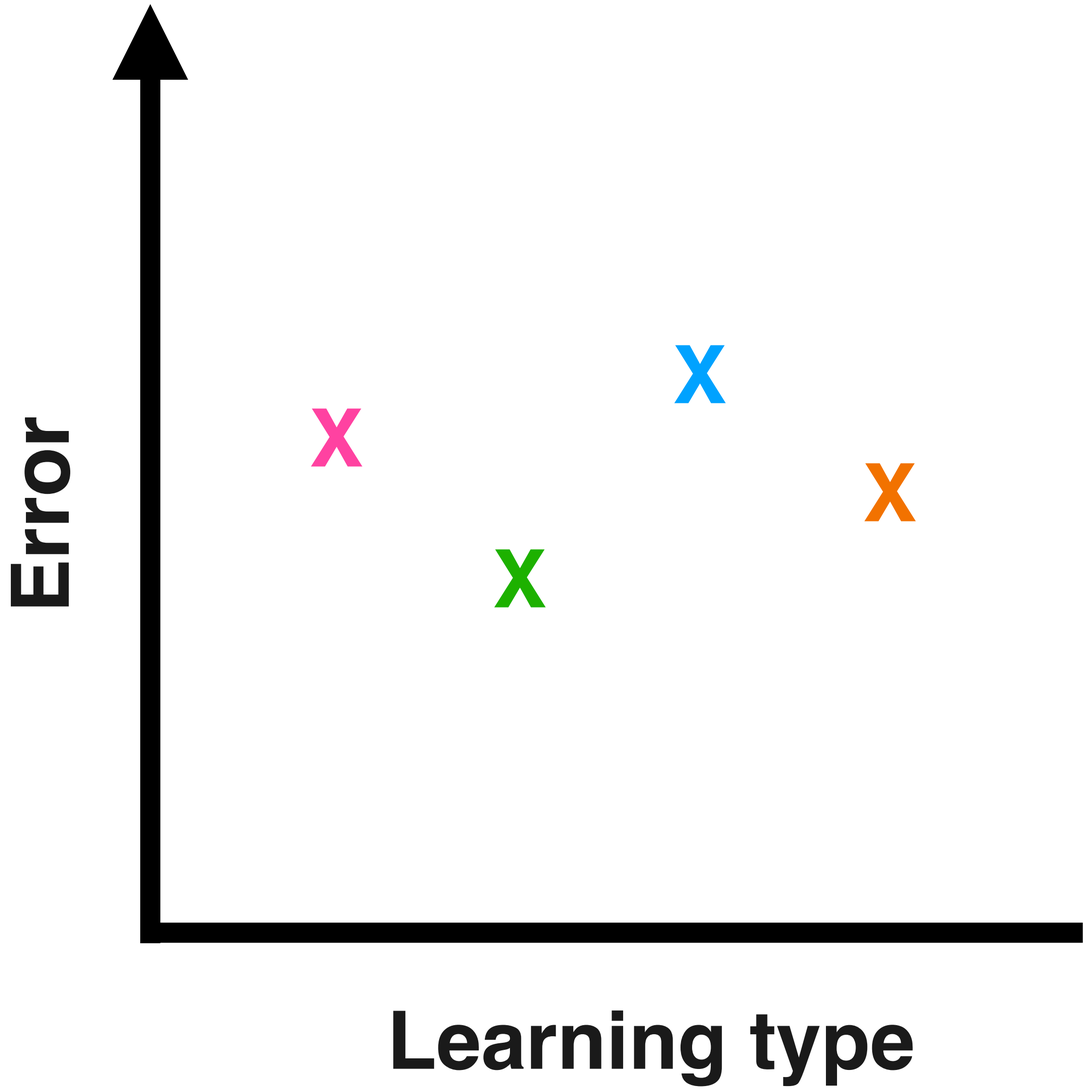
Indexing	Encryption	Compression	Logging	…	Time
TRUE	FALSE	FALSE	FALSE	…	20
TRUE	FALSE	FALSE	TRUE	…	23
TRUE	FALSE	TRUE	FALSE	…	20
TRUE	FALSE	TRUE	TRUE	…	23
TRUE	TRUE	FALSE	FALSE	…	20
TRUE	TRUE	FALSE	TRUE	…	23
TRUE	TRUE	TRUE	FALSE	…	29
TRUE	TRUE	TRUE	TRUE	…	32
FALSE	FALSE	FALSE	FALSE	…	25
…	…	…	…	…	…

*Equal number of samples

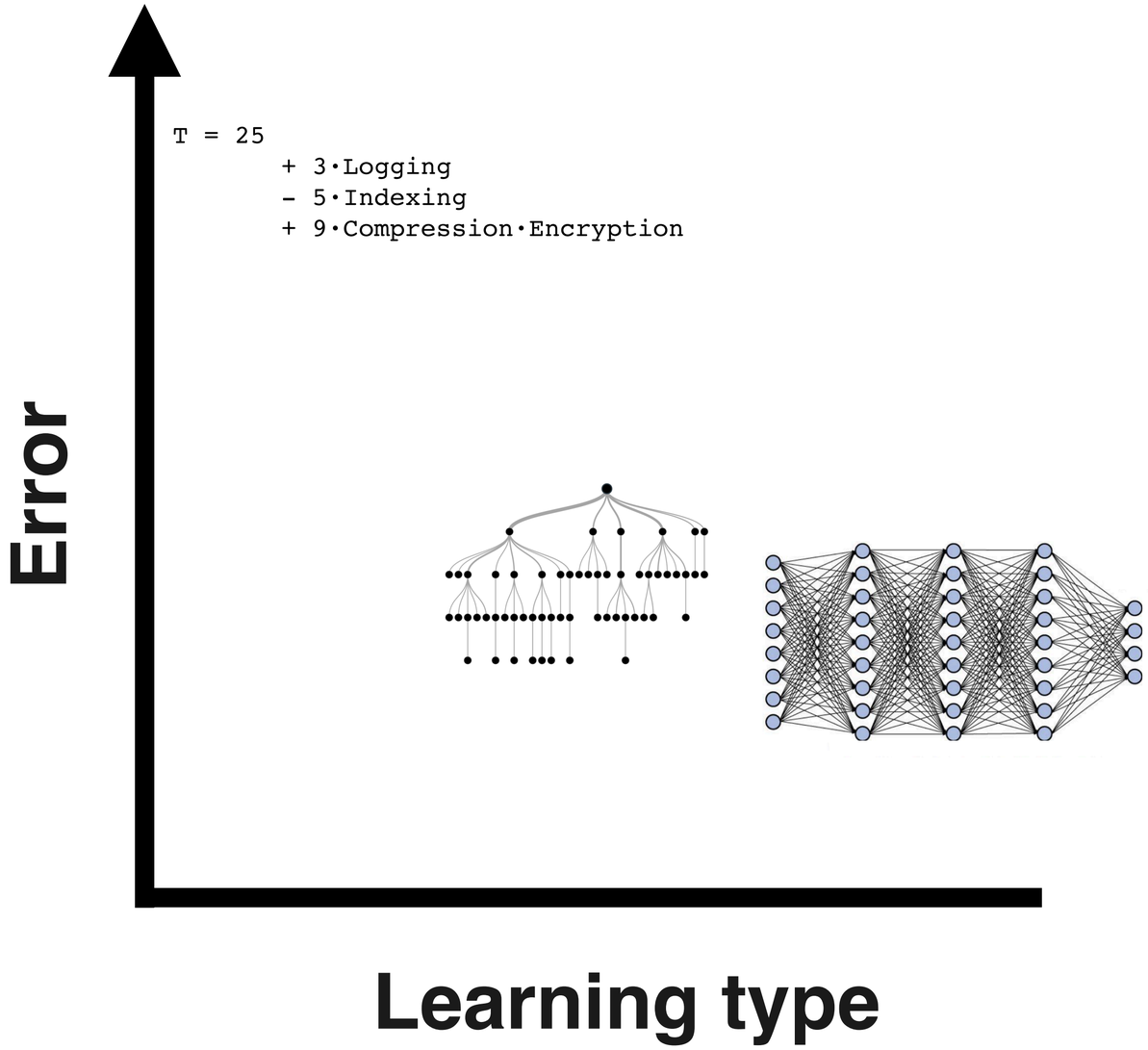


Tradeoff between learning algorithm and accuracy

***Equal number of samples**



***Equal number of samples**



*Equal number of samples

Accurate predictions

*Equal number of samples

Error

Learning type

+ 5·Indexing
+ 9·Compression·Encryption

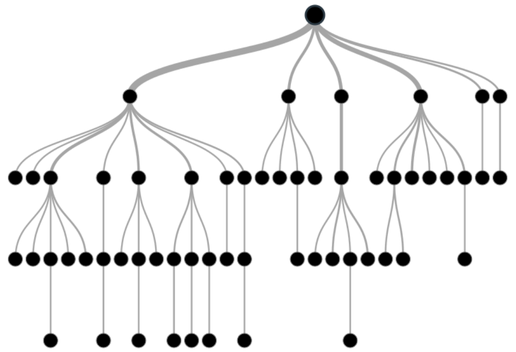
Accurate predictions

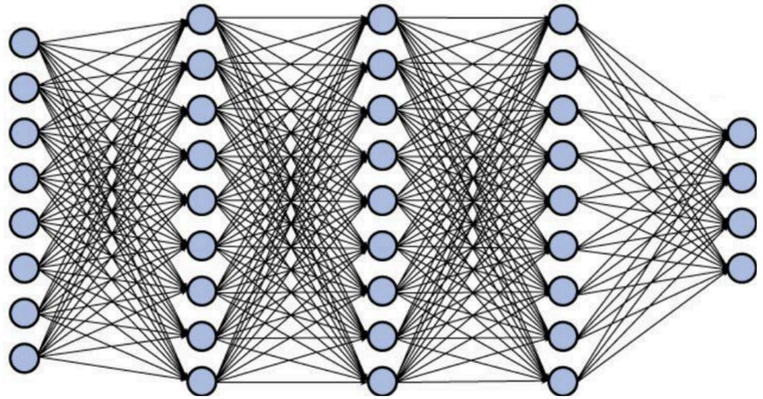
Understanding and debugging

Inherently interpretable

```
T = 25
    + 3·Logging
    - 5·Indexing
    + 9·Compression·Encryption
```

Not inherently interpretable





$$T = 25 + 3 \cdot \text{Logging} - 5 \cdot \text{Caching} \cdot \text{Interrupt} + 9 \cdot \text{Sequential}$$

User

- Make informed tradeoff decisions
- Run systems efficiently

Developer

- Debug surprising behavior
- Develop and maintain efficient software

Understand how options affect performance

```
T = 25
    + 3·Logging
    - 5·Indexing
    + 9·Compression·Encryption
```

User

System

```
T = 25
    + 3·Logging
    - 5·Indexing
    + 9·Compression·Encryption
```

User

System

Make trade off decisions

Performance

Execution time

Energy consumption

Operational costs

```
T = 25
    + 3·Logging
    - 5·Indexing
    + 9·Compression·Encryption
```

Developer

T = 25
+ 3·Logging
- 5·Indexing
·Encryption

But

Developer

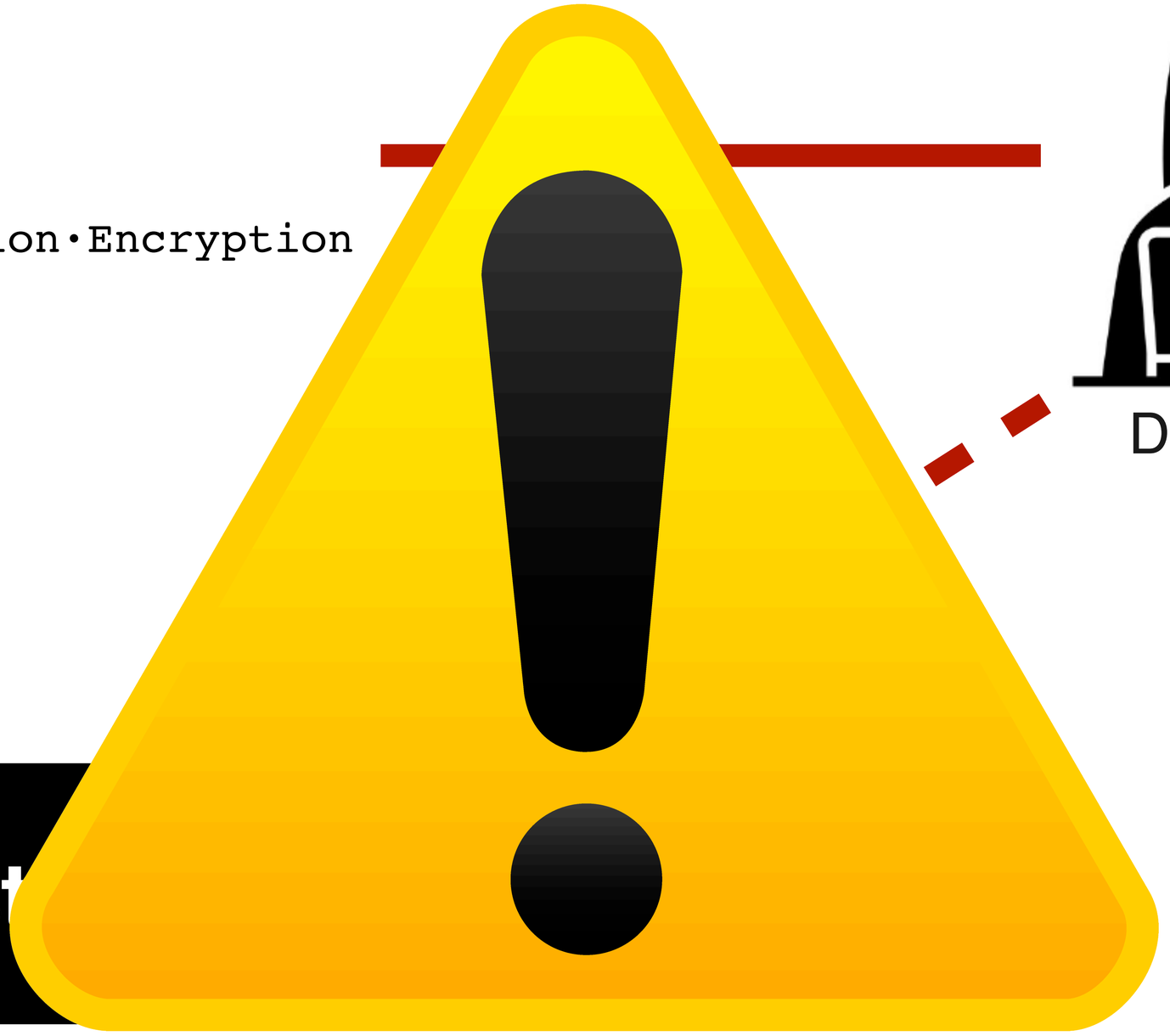
```
T = 25
    + 3·Logging
    - 5·Indexing
    + 9·Compression·Encryption
```

Developer

System

```
T = 25
    + 3·Logging
    - 5·Indexing
    + 9·Compression·Encryption
```

Developer

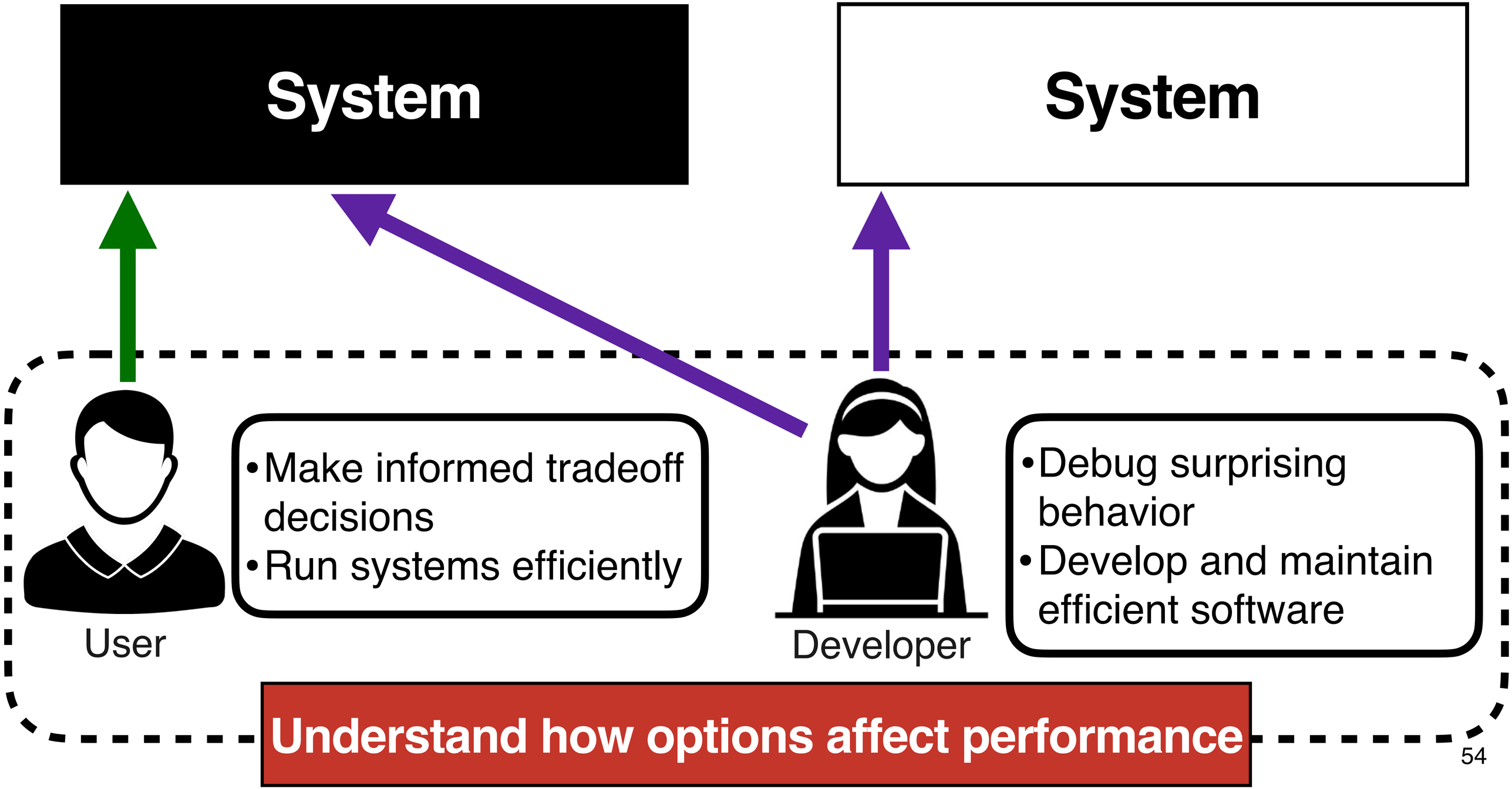


Syst

```
T = 25
    + 3·Logging
    - 5·Indexing
    + 9·Compression·Encryption
```

Developer

System


System
System
User
• Make informed tradeoff decisions
• Run systems efficiently
Developer
• Debug surprising behavior
• Develop and maintain efficient software
Understand how options affect performance

Thesis Statement

White-box analysis of how options influence the performance of code-level structures in configurable systems:

(1) helps to **efficiently build accurate and interpretable global and local performance-influence models**

(2) guides developers to **inspect, understand, and debug configuration-related performance behaviors**.

Thesis Statement

White-box analysis of how options influence the performance of code-level structures in configurable systems:

(1) helps to **efficiently build accurate and interpretable global and local performance-influence models**

(2) guides developers to **inspect, understand, and debug configuration-related performance behaviors**.

Thesis Statement

White-box analysis of how options influence the performance of code-level structures in configurable systems:

(1) helps to **efficiently build accurate and interpretable global and local performance-influence models**

(2) guides developers to **inspect, understand, and debug configuration-related performance behaviors**.

Thesis Statement

White-box analysis of how options influence the performance of code-level structures in configurable systems:

(1) helps to **efficiently build accurate and interpretable global and local performance-influence models**

(2) guides developers to **inspect, understand, and debug configuration-related performance behaviors**.

Thesis Statement

White-box analysis of how options influence the performance of **code-level structures** in configurable systems:

(1) helps to **efficiently build accurate and interpretable global and local performance-influence models**

(2) guides developers to **inspect, understand, and debug configuration-related performance behavior**.

But why white-box?

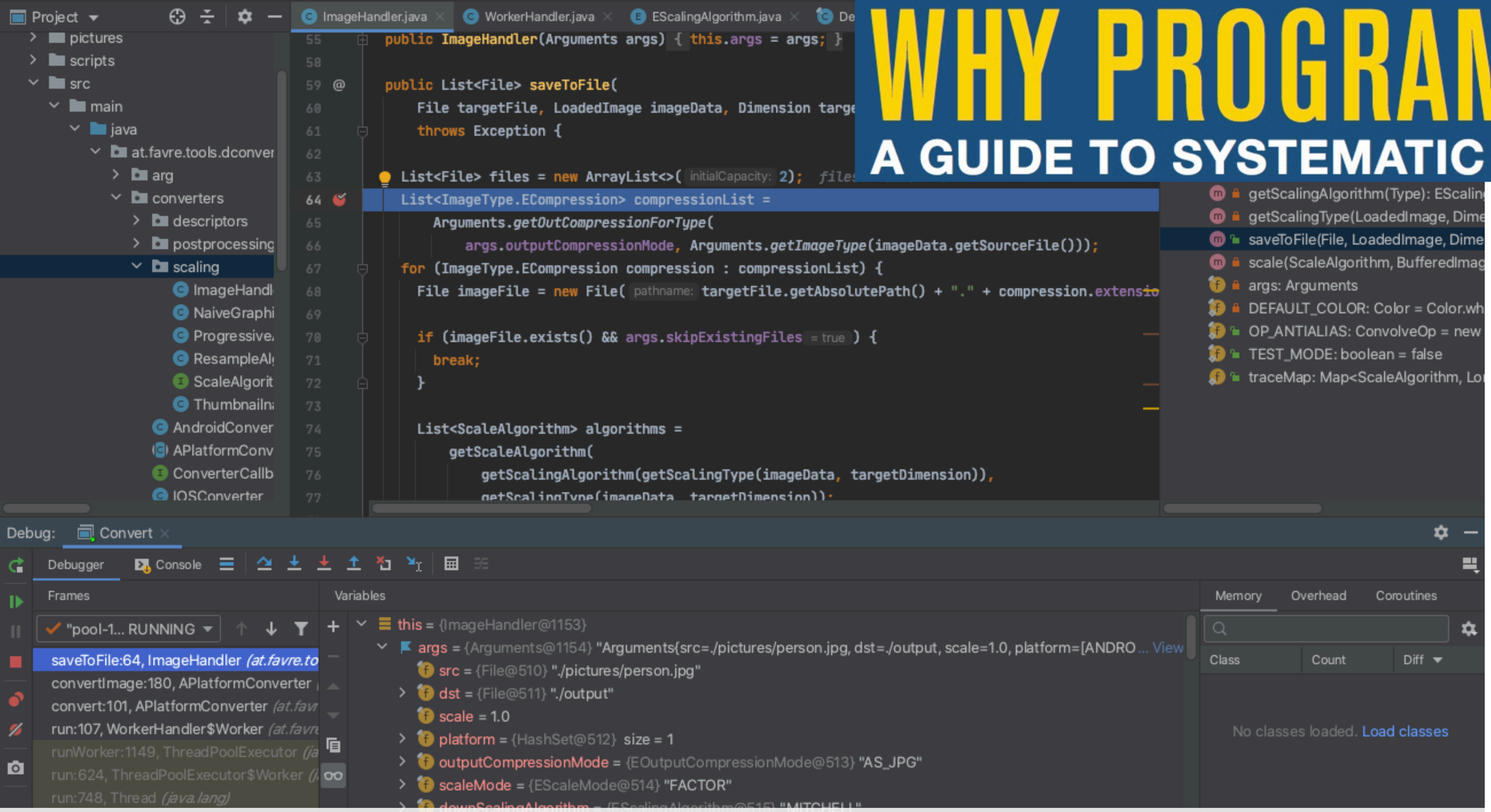
Insights!

Not all options tend to affect the performance for all workloads

Few options tend to interact in configurable systems

Insight!

Performance in configurable systems tends to change at control-flow statements



Debugging requires analyzing the implementation

User

- Make informed tradeoff decisions
- Run systems efficiently

Developer

- Debug surprising behavior
- Develop and maintain efficient software

Understand how options affect performance

1 Modeling

Global	Local

User

- Make informed tradeoff decisions
- Run systems efficiently

Developer

- Debug surprising behavior
- Develop and maintain efficient software

Understand how options affect performance

1 **Modeling**

Global | **Local**

2 **Debugging**

User

- Make informed tradeoff decisions
- Run systems efficiently

Developer

- Debug surprising behavior
- Develop and maintain efficient software

Understand how options affect performance

1 Modeling

Global | Local

2 Debugging

User

- Make informed tradeoff decisions
- Run systems efficiently

Developer

- Debug surprising behavior
- Develop and maintain efficient software

Understand how options affect performance

1 Modeling

Global | Local

2 Debugging

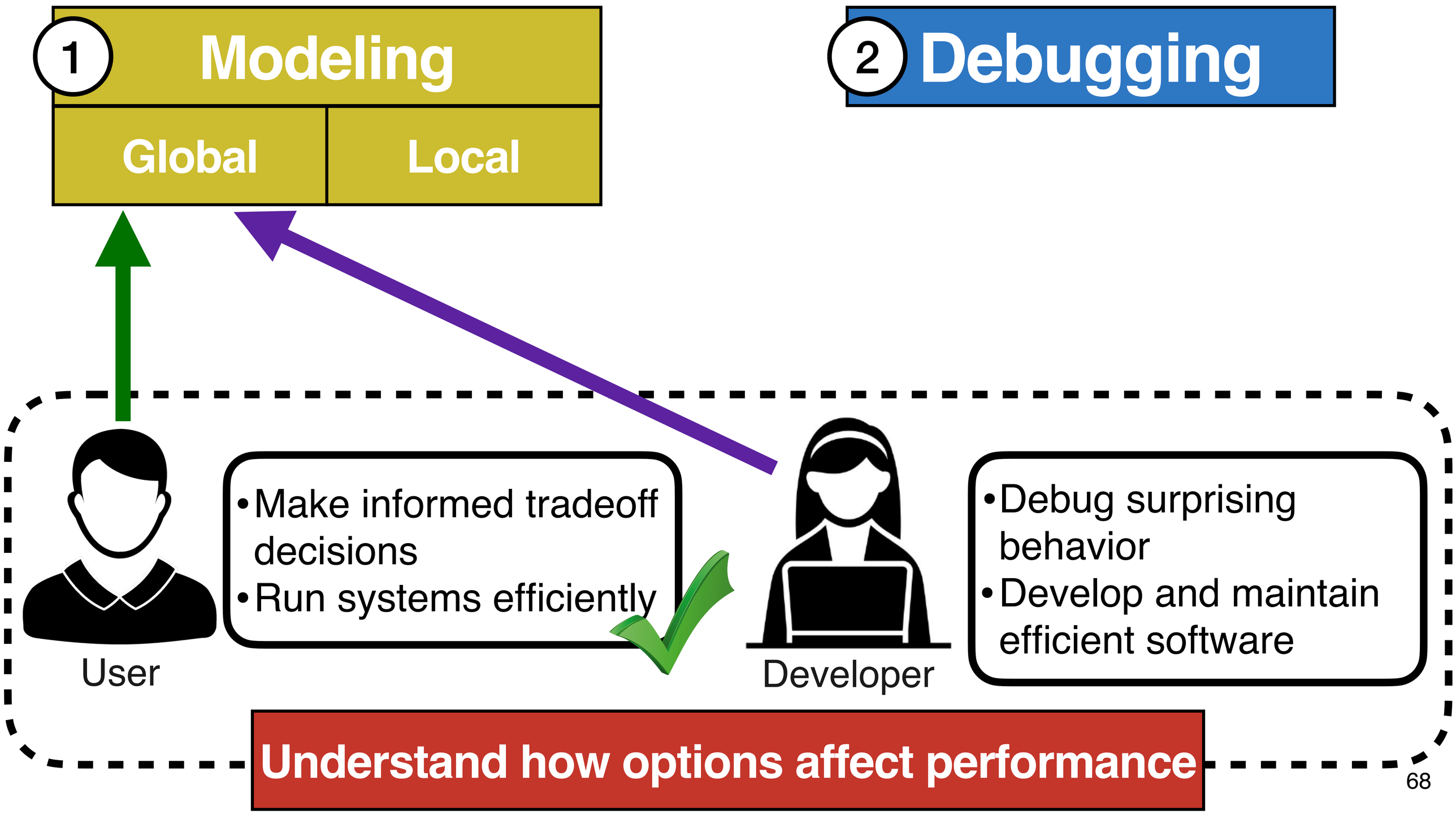
User

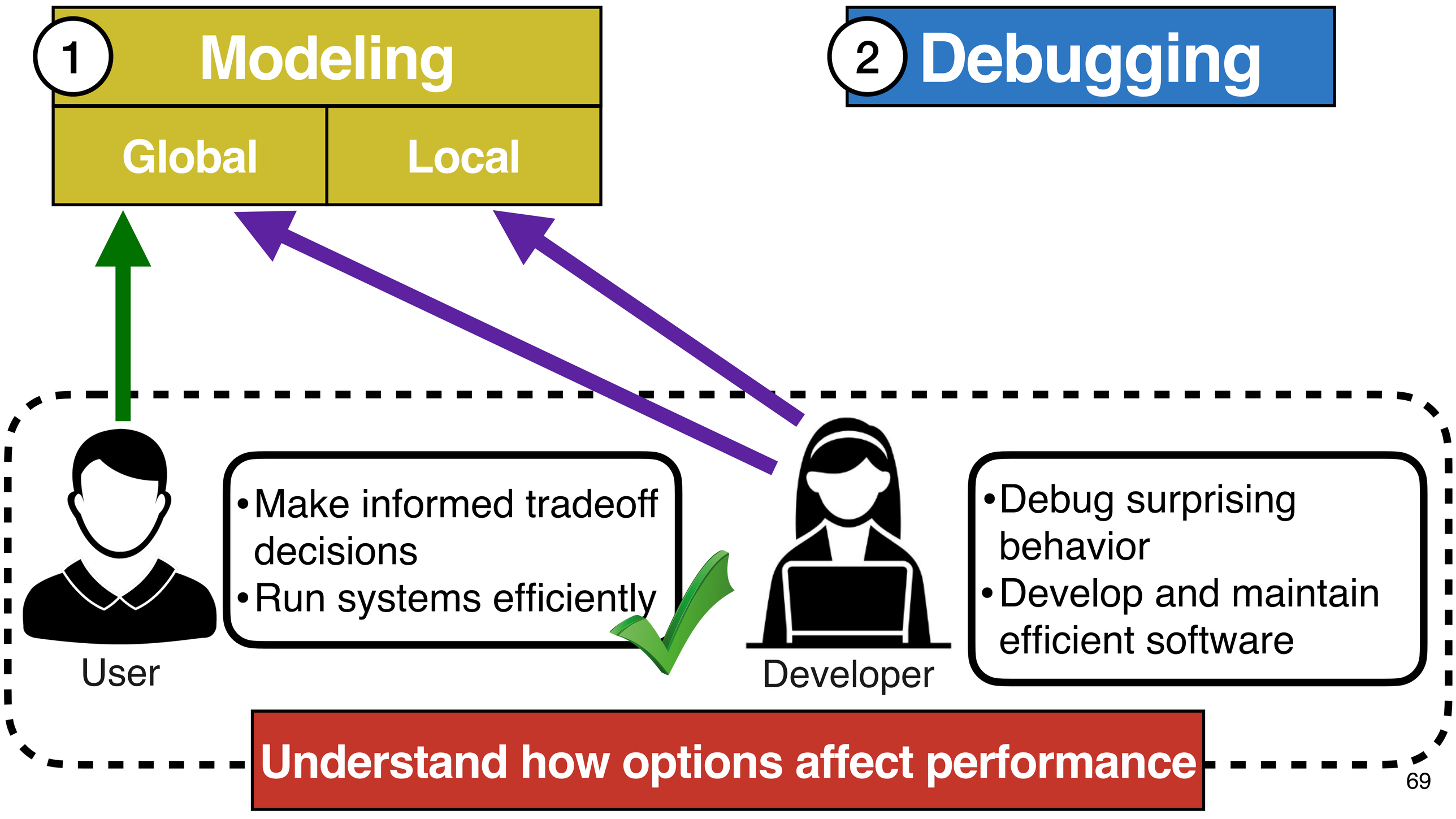
- Make informed tradeoff decisions
- Run systems efficiently

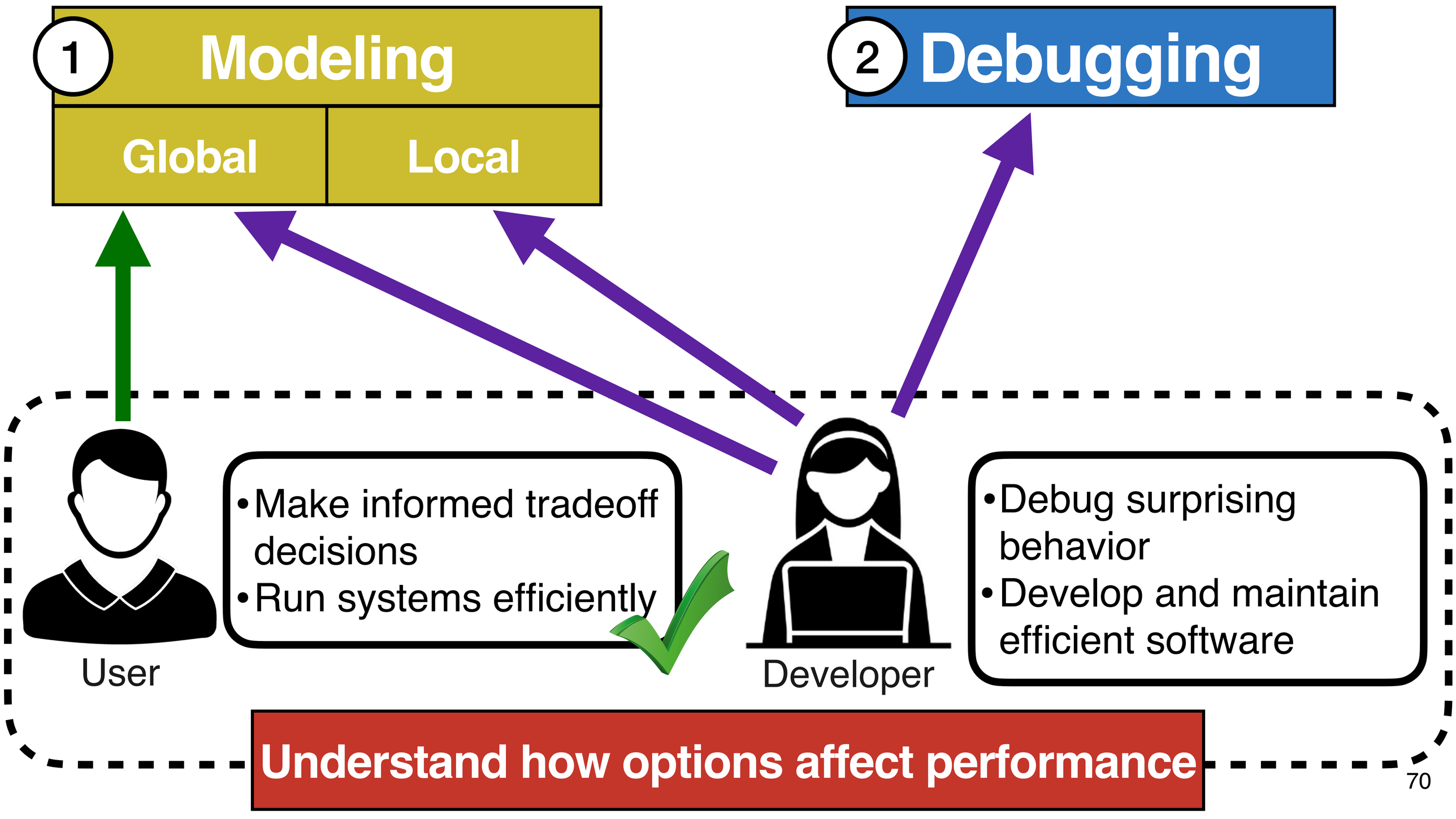
Developer

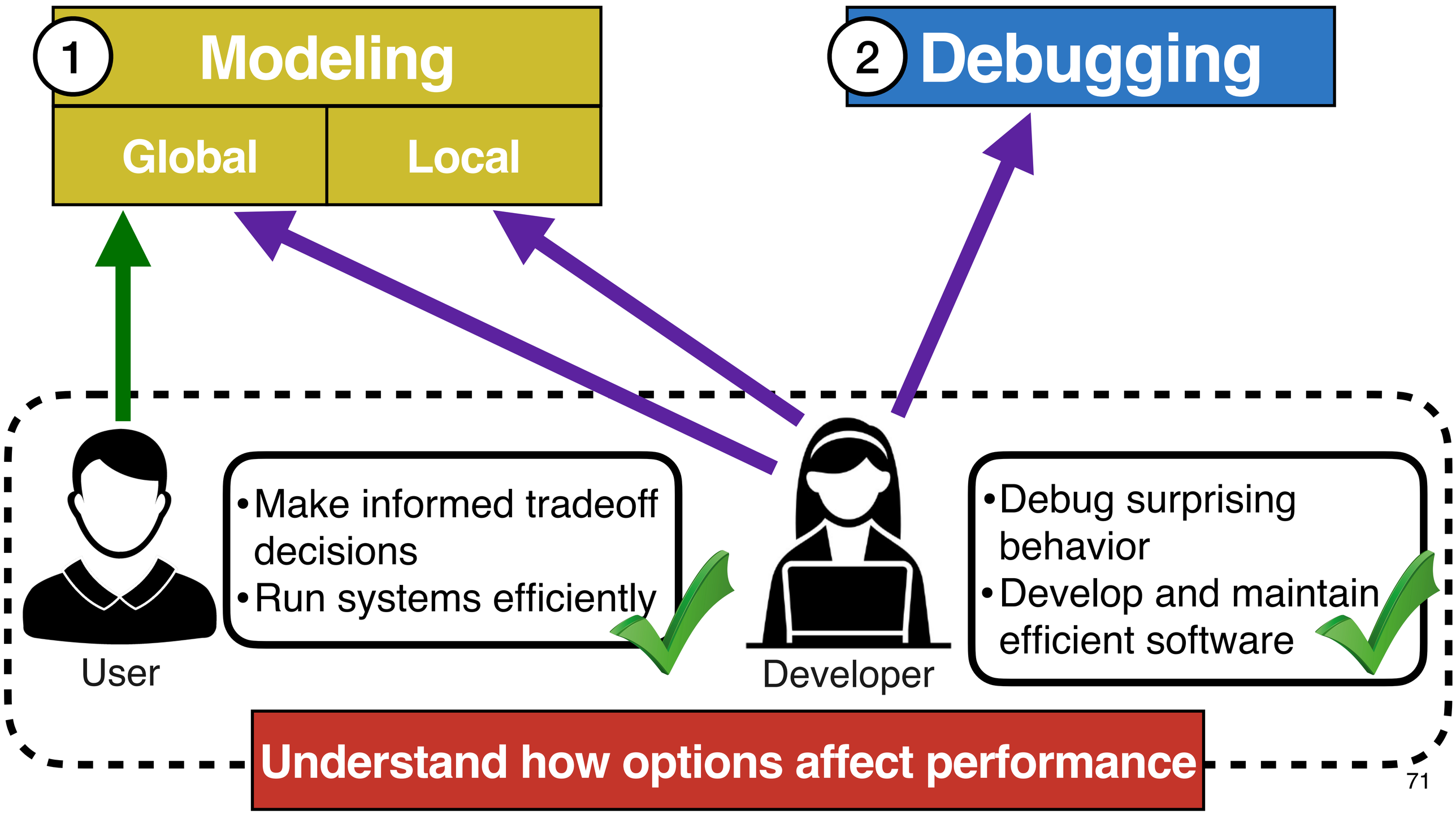
- Debug surprising behavior
- Develop and maintain efficient software

Understand how options affect performance

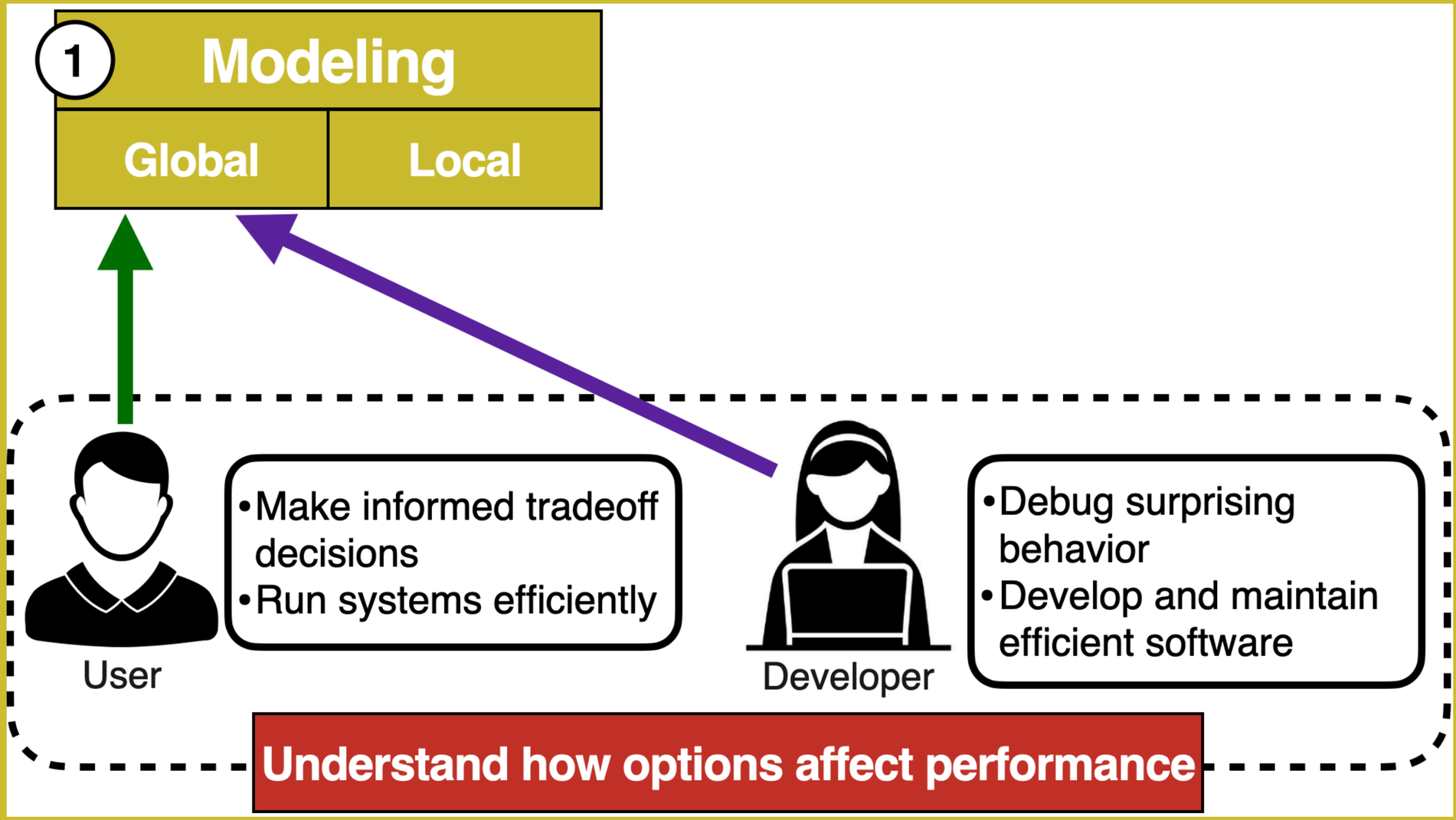

1 Modeling
Global
Local
2 Debugging
User
• Make informed tradeoff decisions
• Run systems efficiently
Developer
• Debug surprising behavior
• Develop and maintain efficient software
Understand how options affect performance


1
Modeling
Global
Local
2
Debugging
User
•Make informed tradeoff decisions
•Run systems efficiently
Developer
•Debug surprising behavior
•Develop and maintain efficient software
Understand how options affect performance


1
Modeling
Global
Local
2
Debugging
User
•Make informed tradeoff decisions
•Run systems efficiently
Developer
•Debug surprising behavior
•Develop and maintain efficient software
Understand how options affect performance


1
Modeling
Global
Local
2
Debugging
User
• Make informed tradeoff decisions
• Run systems efficiently
Developer
• Debug surprising behavior
• Develop and maintain efficient software
Understand how options affect performance

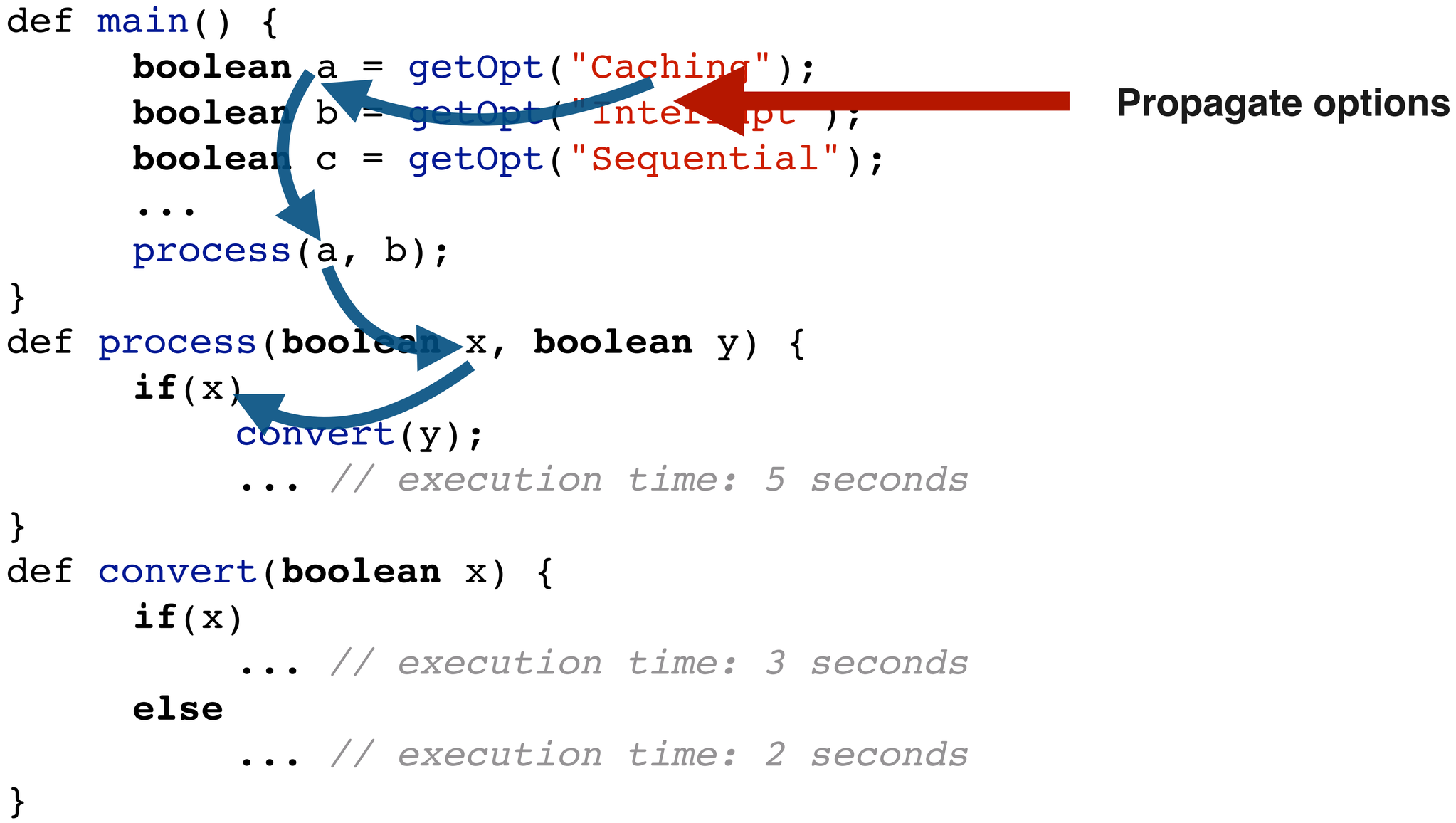
White-box Performance Modeling of Configurable Systems



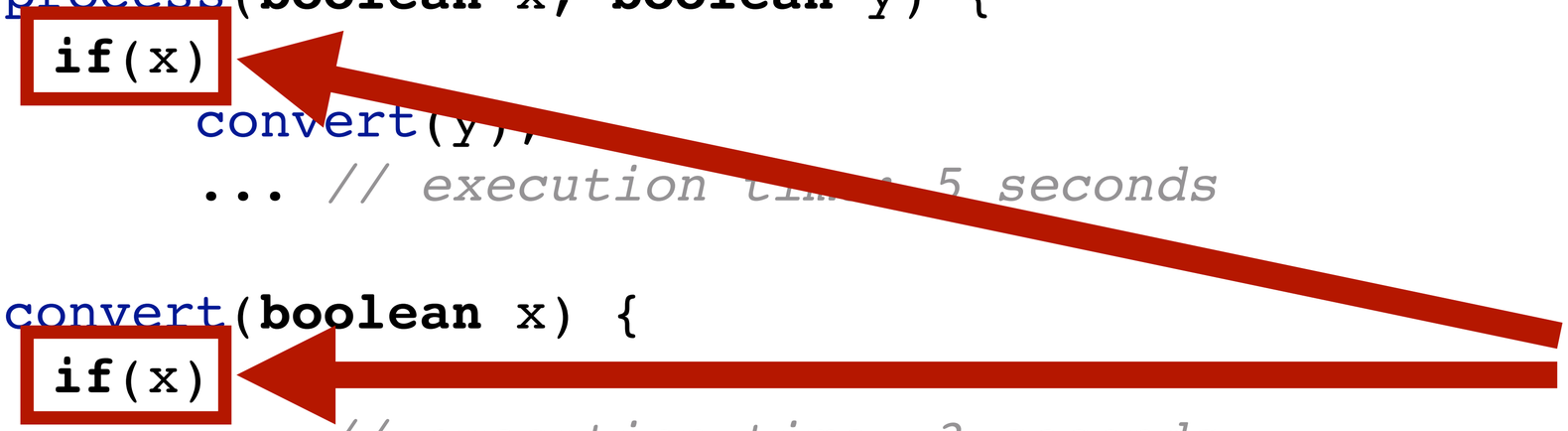
```
1  def main() {
2        boolean a = getOpt("Caching");
3        boolean b = getOpt("Interrupt");
4        boolean c = getOpt("Sequential");
5        ...
6        process(a, b);
7  }
8  def process(boolean x, boolean y) {
9        if(x)
10           convert(y);
11           ... // execution time: 5 seconds
12 }
13 def convert(boolean x) {
14       if(x)
15           ... // execution time: 3 seconds
16       else
17           ... // execution time: 2 seconds
18 }
```

```
1  def main() {
2        boolean a = getOpt("Caching");
3        boolean b = getOpt("Interrupt");
4        boolean c = getOpt("Sequential");
5        ...
6        process(a, b);
7  }
8  def process(boolean x, boolean y) {
9        if(x)
10           convert(y);
11           ... // execution time: 5 seconds
12 }
13 def convert(boolean x) {
14       if(x)
15           ... // execution time: 3 seconds
16       else
17           ... // execution time: 2 seconds
18 }
```

Load options


1 def main() {
2 boolean a = getOpt("Caching");
3 boolean b = getOpt("Interrupt");
4 boolean c = getOpt("Sequential");
5 ...
6 process(a, b);
7 }
8 def process(boolean x, boolean y) {
9 if(x)
10 convert(y);
11 ... // execution time: 5 seconds
12 }
13 def convert(boolean x) {
14 if(x)
15 ... // execution time: 3 seconds
16 else
17 ... // execution time: 2 seconds
18 }
Propagate options

```
1  def main() {
2        boolean a = getOpt("Caching");
3        boolean b = getOpt("Interrupt");
4        boolean c = getOpt("Sequential");
5        ...
6        process(a, b);
7  }
8  def process(boolean x, boolean y) {
9        if(x)
10             convert(y);
11             ... // execution time: 5 seconds
12 }
13 def convert(boolean x) {
14       if(x)
15             ... // execution time: 3 seconds
16       else
17             ... // execution time: 2 seconds
18 }
```



Use options

```
1  def main() {
2        boolean a = getOpt("Caching");
3        boolean b = getOpt("Interrupt");
4        boolean c = getOpt("Sequential");
5        ...
6        process(a, b);
7  }
8  def process(boolean x, boolean y) {
9        if(x)
10           convert(y);
11           ... // execution time: 5 seconds
12 }
13 def convert(boolean x) {
14       if(x)
15           ... // execution time: 3 seconds
16       else
17           ... // execution time: 2 seconds
18 }
```

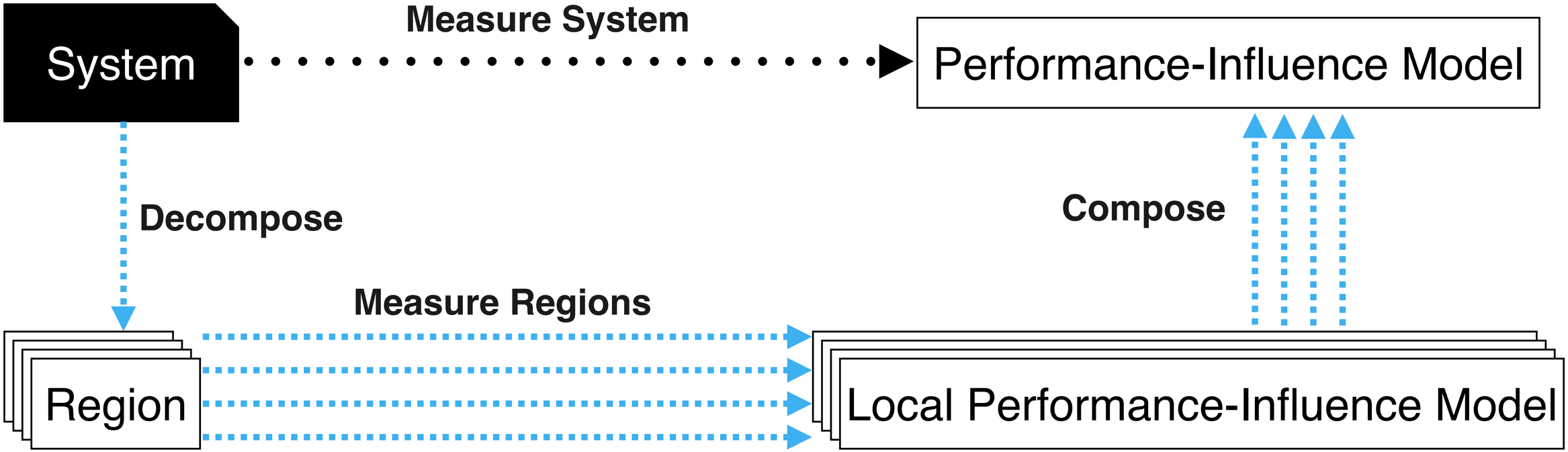
Our insights!

Compositionality

Compression

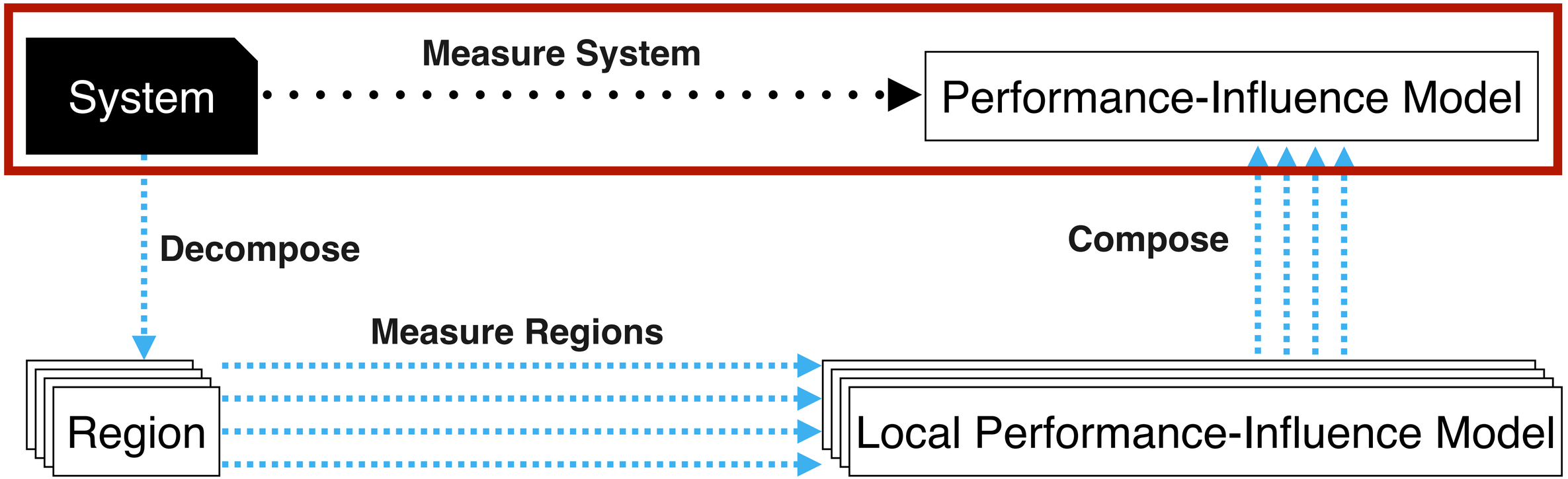
Compositionality

Performance-influence models can be built by composing models built independently for smaller regions of code



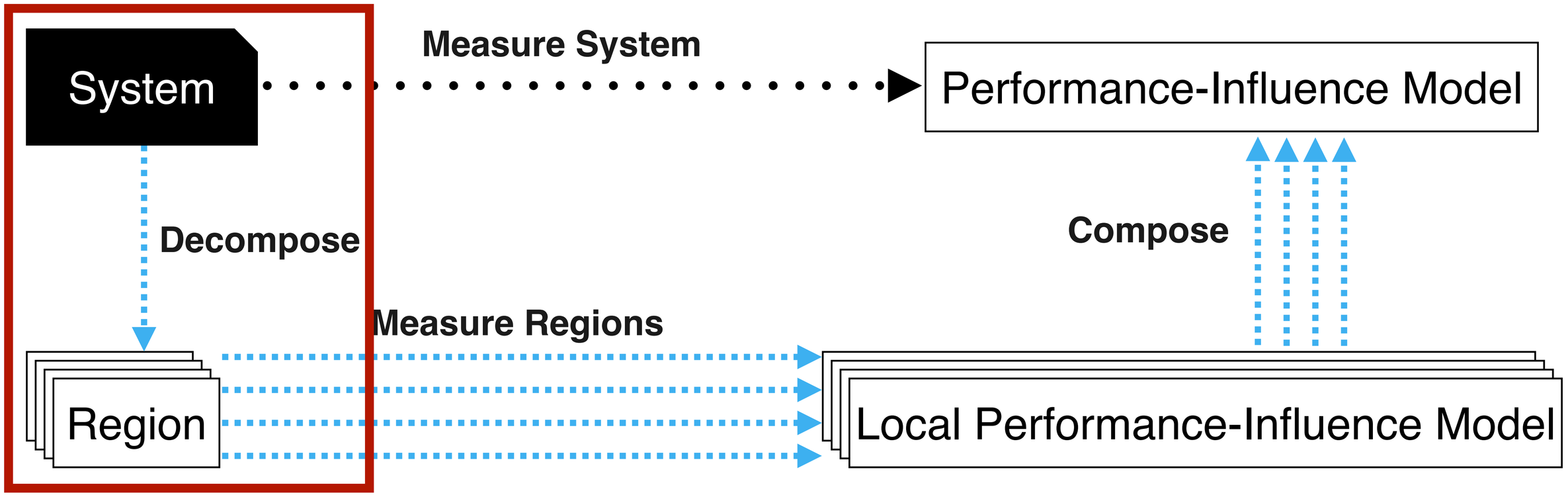
Compositionality

Performance-influence models can be built by composing models built independently for smaller regions of code



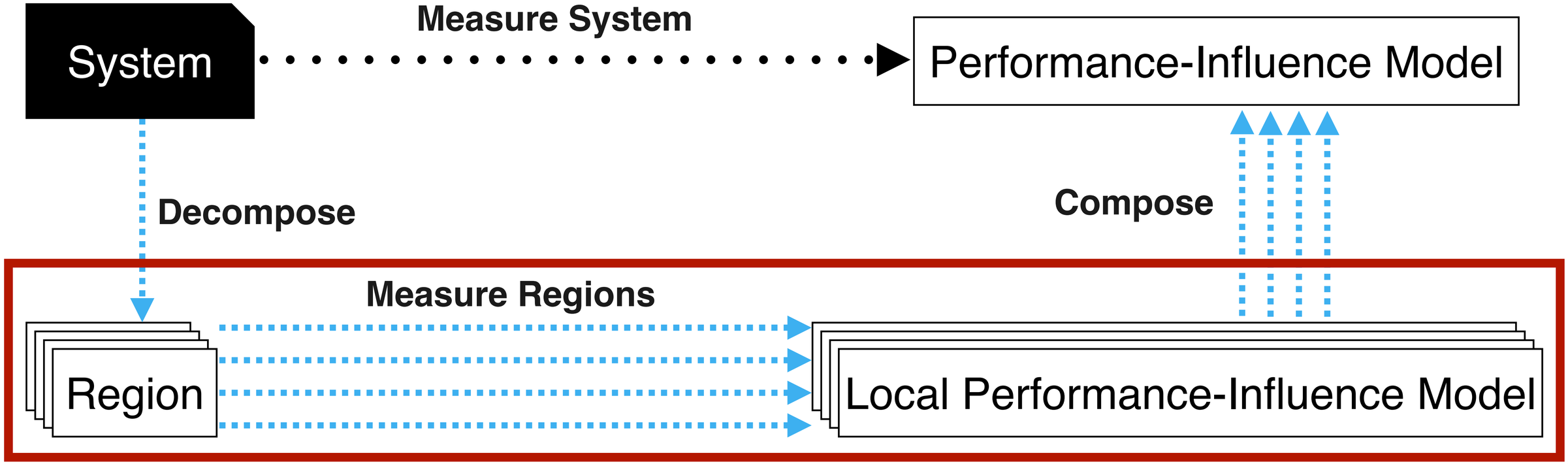
Compositionality

Performance-influence models can be built by composing models built independently for smaller regions of code



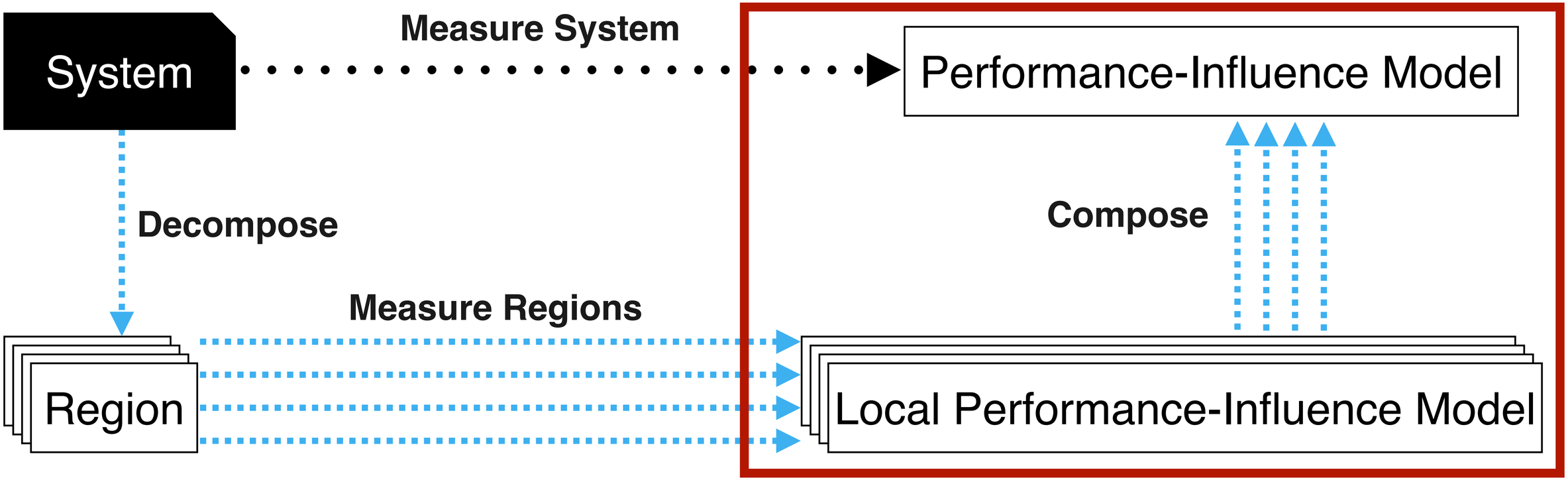
Compositionality

Performance-influence models can be built by composing models built independently for smaller regions of code



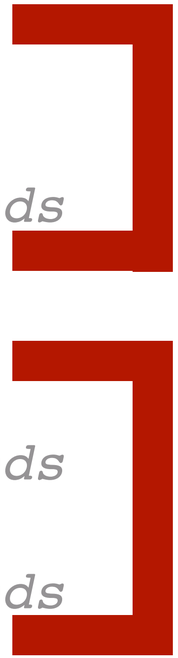
Compositionality

Performance-influence models can be built by composing models built independently for smaller regions of code



```
1  def main() {
2        boolean a = getOpt("Caching");
3        boolean b = getOpt("Interrupt");
4        boolean c = getOpt("Sequential");
5        ...
6        process(a, b);
7  }
8  def process(boolean x, boolean y) {
9        if(x)
10           convert(y);
11           ... // execution time: 5 seconds
12 }
13 def convert(boolean x) {
14       if(x)
15           ... // execution time: 3 seconds
16       else
17           ... // execution time: 2 seconds
18 }
```

```
1  def main() {
2        boolean a = getOpt("Caching");
3        boolean b = getOpt("Interrupt");
4        boolean c = getOpt("Sequential");
5        ...
6        process(a, b);
7  }
8  def process(boolean x, boolean y) {
9        if(x)
10           convert(y);
11           ... // execution time: 5 seconds
12 }
13 def convert(boolean x) {
14       if(x)
15           ... // execution time: 3 seconds
16       else
17           ... // execution time: 2 seconds
18 }
```



```
1  def main() {
2        boolean a = getOpt("Caching");
3        boolean b = getOpt("Interrupt");
4        boolean c = getOpt("Sequential");
5        ...
6        process(a, b);
7  }
8  def process(boolean x, boolean y) {
9        if(x) // region depends on Caching
10           convert(y);
11           ... // execution time: 5 seconds
12 }
13 def convert(boolean x) {
14       if(x) // region depends on Interrupt
15           ... // execution time: 3 seconds
16       else
17           ... // execution time: 2 seconds
18 }
```

Caching	Time
TRUE	
FALSE	

```
1  def main() {
2        boolean a = getOpt("Caching");
3        boolean b = getOpt("Interrupt");
4        boolean c = getOpt("Sequential");
5        ...
6        process(a, b);
7  }
8  def process(boolean x, boolean y) {
9        if(x) // region depends on Caching
10           convert(y);
11           ... // execution time: 5 seconds
12 }
13 def convert(boolean x) {
14       if(x) // region depends on Interrupt
15           ... // execution time: 3 seconds
16       else
17           ... // execution time: 2 seconds
18 }
```

Caching	Time
TRUE	5
FALSE	

```
1  def main() {
2        boolean a = getOpt("Caching");
3        boolean b = getOpt("Interrupt");
4        boolean c = getOpt("Sequential");
5        ...
6        process(a, b);
7  }
8  def process(boolean x, boolean y) {
9        if(x) // region depends on Caching
10           convert(y);
11           ... // execution time: 5 seconds
12 }
13 def convert(boolean x) {
14       if(x) // region depends on Interrupt
15           ... // execution time: 3 seconds
16       else
17           ... // execution time: 2 seconds
18 }
```

Caching	Time
TRUE	5
FALSE	0

```
1  def main() {
2        boolean a = getOpt("Caching");
3        boolean b = getOpt("Interrupt");
4        boolean c = getOpt("Sequential");
5        ...
6        process(a, b);
7  }
8  def process(boolean x, boolean y) {
9        if(x) // region depends on Caching
10           convert(y);
11           ... // execution time: 5 seconds
12 }
13 def convert(boolean x) {
14       if(x) // region depends on Interrupt
15           ... // execution time: 3 seconds
16       else
17           ... // execution time: 2 seconds
18 }
```

Caching	Time
TRUE	5
FALSE	0

$T_{process} = 5 \cdot Caching$

```
1  def main() {
2        boolean a = getOpt("Caching");
3        boolean b = getOpt("Interrupt");
4        boolean c = getOpt("Sequential");
5        ...
6        process(a, b);
7  }
8  def process(boolean x, boolean y) {
9        if(x) // region depends on Caching
10           convert(y);
11           ... // execution time: 5 seconds
12 }
13 def convert(boolean x) {
14       if(x) // region depends on Interrupt
15           ... // execution time: 3 seconds
16       else
17           ... // execution time: 2 seconds
18 }
```

$T_{process} = 5 \cdot Caching$

```
1  def main() {
2        boolean a = getOpt("Caching");
3        boolean b = getOpt("Interrupt");
4        boolean c = getOpt("Sequential");
5        ...
6        process(a, b);
7  }
8  def process(boolean x, boolean y) {
9        if(x) // region depends on Caching
10           convert(y);
11           ... // execution time: 5 seconds
12 }
13 def convert(boolean x) {
14       if(x) // region depends on Interrupt
15           ... // execution time: 3 seconds
16       else
17           ... // execution time: 2 seconds
18 }
```

$T_{process} = 5 \cdot Caching$

Caching	Interrupt	Time
TRUE	TRUE	3
TRUE	FALSE	2

$T_{convert} = 2 \cdot Caching + 1 \cdot Caching \cdot Interrupt$

```
1  def main() {
2        boolean a = getOpt("Caching");
3        boolean b = getOpt("Interrupt");
4        boolean c = getOpt("Sequential");
5        ...
6        process(a, b);
7  }
8  def process(boolean x, boolean y) {
9        if(x) // region depends on Caching
10           convert(y);
11           ... // execution time: 5 seconds
12 }
13 def convert(boolean x) {
14       if(x) // region depends on Interrupt
15           ... // execution time: 3 seconds
16       else
17           ... // execution time: 2 seconds
18 }
```

$T_{process} = 5 \cdot Caching$

$T_{convert} = 2 \cdot Caching + 1 \cdot Caching \cdot Interrupt$

Compositionality

```
1  def main() {
2        boolean a = getOpt("Caching");
3        boolean b = getOpt("Interrupt");
4        boolean c = getOpt("Sequential");
5        ...
6        process(a, b);
7  }
8  def process(boolean x, boolean y) {
9        if(x) // region depends on Caching
10           convert(y);
11           ... // execution time: 5 seconds
12 }
13 def convert(boolean x) {
14       if(x) // region depends on Interrupt
15           ... // execution time: 3 seconds
16       else
17           ... // execution time: 2 seconds
18 }
```

$T_{process} = 5 \cdot Caching$

+

$T_{convert} = 2 \cdot Caching + 1 \cdot Caching \cdot Interrupt$

Compositionality

```
1  def main() {
2        boolean a = getOpt("Caching");
3        boolean b = getOpt("Interrupt");
4        boolean c = getOpt("Sequential");
5        ...
6        process(a, b);
7  }
8  def process(boolean x, boolean y) {
9        if(x) // region depends on Caching
10           convert(y);
11           ... // execution time: 5 seconds
12 }
13 def convert(boolean x) {
14       if(x) // region depends on Interrupt
15           ... // execution time: 3 seconds
16       else
17           ... // execution time: 2 seconds
18 }
```

$T_{process} = 5 \cdot Caching$

+

$T_{convert} = 2 \cdot Caching + 1 \cdot Caching \cdot Interrupt$

$$T = 7 \cdot Caching + 1 \cdot Caching \cdot Interrupt$$

Insight!

Compositionality

$T_{process} = 5 \cdot Caching$

+

$T_{convert} = 2 \cdot Caching + 1 \cdot Caching \cdot Interrupt$

Few options tend to interact in smaller regions

$T = 7 \cdot Caching + 1 \cdot Caching \cdot Interrupt$

Our insights!

Compositionality

Compression

Compression

Compression allow us to simultaneously explore paths in multiple independent regions with a few configurations

Compression

Caching	Interrupt	Sequential

```
1  def main() {
2        boolean a = getOpt(“Caching");
3        boolean b = getOpt("Interrupt");
4        boolean c = getOpt("Sequential");
5        ...
6        if(a) // variable depends on Caching
7             ... // execution time: 1 second


8        if(b) // variable depends on Interrupt
9             ... // execution time: 2 seconds


10       if(c) // variable depends on Sequential
11            ... // execution time: 3 seconds
12 }
```

Caching	Time
TRUE	
FALSE	

Interrupt	Time
TRUE	
FALSE	

Sequential	Time
TRUE	
FALSE	

Compression

Caching	Interrupt	Sequential
FALSE	FALSE	FALSE

```
1  def main() {
2        boolean a = getOpt(“Caching");
3        boolean b = getOpt("Interrupt");
4        boolean c = getOpt("Sequential");
5        ...
6        if(a) // variable depends on Caching
7             ... // execution time: 1 second


8        if(b) // variable depends on Interrupt
9             ... // execution time: 2 seconds


10       if(c) // variable depends on Sequential
11            ... // execution time: 3 seconds
12 }
```

Caching	Time
TRUE	
FALSE	0

Interrupt	Time
TRUE	
FALSE	0

Sequential	Time
TRUE	
FALSE	0

Compression

Caching	Interrupt	Sequential
FALSE	FALSE	FALSE
TRUE	TRUE	TRUE

```
1  def main() {
2        boolean a = getOpt(“Caching");
3        boolean b = getOpt("Interrupt");
4        boolean c = getOpt("Sequential");
5        ...
6        if(a) // variable depends on Caching
7             ... // execution time: 1 second


8        if(b) // variable depends on Interrupt
9             ... // execution time: 2 seconds


10       if(c) // variable depends on Sequential
11            ... // execution time: 3 seconds
12 }
```

Caching	Time
TRUE	1
FALSE	0

Interrupt	Time
TRUE	2
FALSE	0

Sequential	Time
TRUE	3
FALSE	0

Compression

Insight!

Caching	Interrupt	Sequential
FALSE	FALSE	FALSE
TRUE	TRUE	TRUE

```
1 de
2     ean a = getOpt("Caching");
3     lean b = getOpt("Interrupt");
4     lean c = getOpt("Sequential");
5
6     if(a) // variable depends on Caching
7         ... // execution time: 1 second
8     if(b) // variable depends on Interrupt
9         ... // execution time: 2 seconds
10    if(c) // variable depends on Sequential
11        ... // execution time: 3 seconds
12 }
```

Caching	Time
TRUE	1
FALSE	0

Interrupt	Time
TRUE	2
FALSE	0

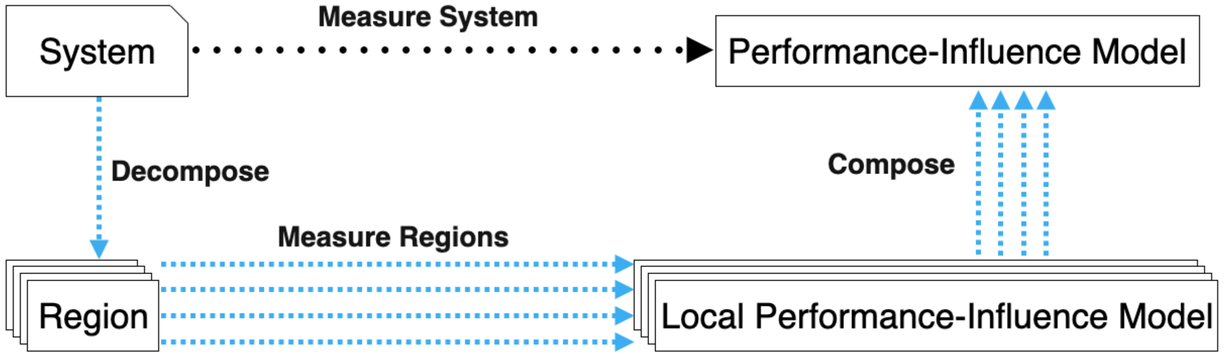
Sequential	Time
TRUE	3
FALSE	0

Few different options tend to interact in different regions

Compositionality

Compression

Our insights!



Caching	Interrupt	Sequential
FALSE	FALSE	FALSE
TRUE	TRUE	TRUE

```
1  def main() {
2        boolean a = getOpt(“Caching");
3        boolean b = getOpt("Interrupt");
4        boolean c = getOpt("Sequential");
5        ...
6        if(a) // variable depends on Caching
7             ... // execution time: 1 second

8        if(b) // variable depends on Interrupt
9             ... // execution time: 2 seconds

10       if(c) // variable depends on Sequential
11            ... // execution time: 3 seconds
12 }
```

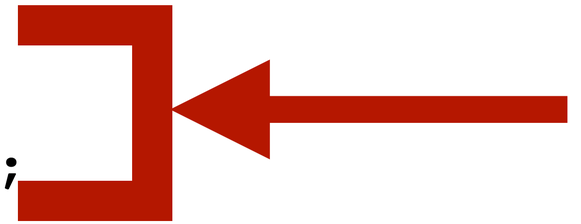
Caching	Time
TRUE	1
FALSE	0

Interrupt	Time
TRUE	2
FALSE	0

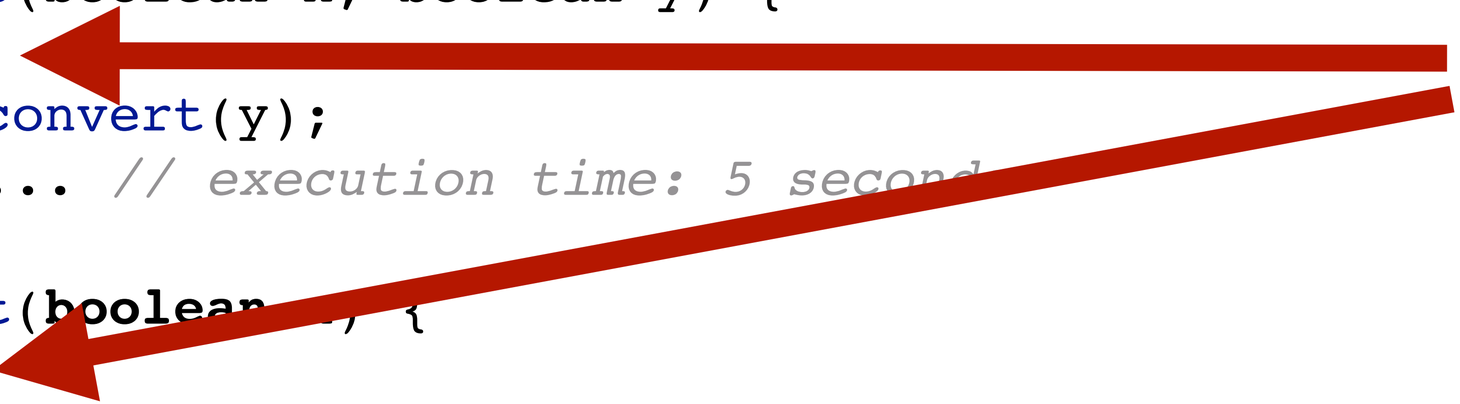
Sequential	Time
TRUE	3
FALSE	0

Taint Analysis

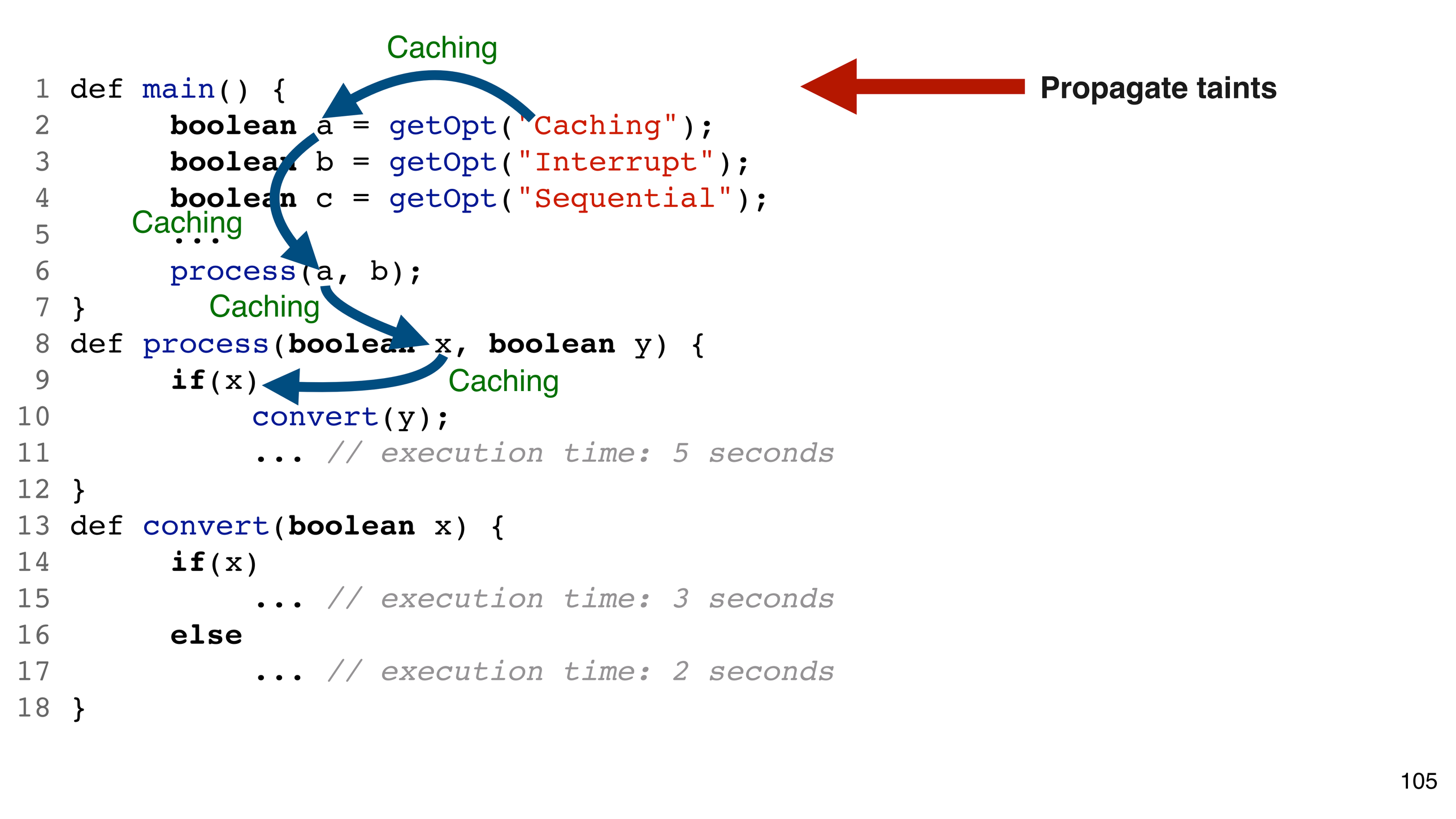
```
1  def main() {
2        boolean a = getOpt("Caching");
3        boolean b = getOpt("Interrupt");
4        boolean c = getOpt("Sequential");
5        ...
6        process(a, b);
7  }
8  def process(boolean x, boolean y) {
9        if(x)
10           convert(y);
11           ... // execution time: 5 second
12 }
13 def convert(boolean x) {
14       if(x)
15           ... // execution time: 3 seconds
16       else
17           ... // execution time: 2 seconds
18 }
```



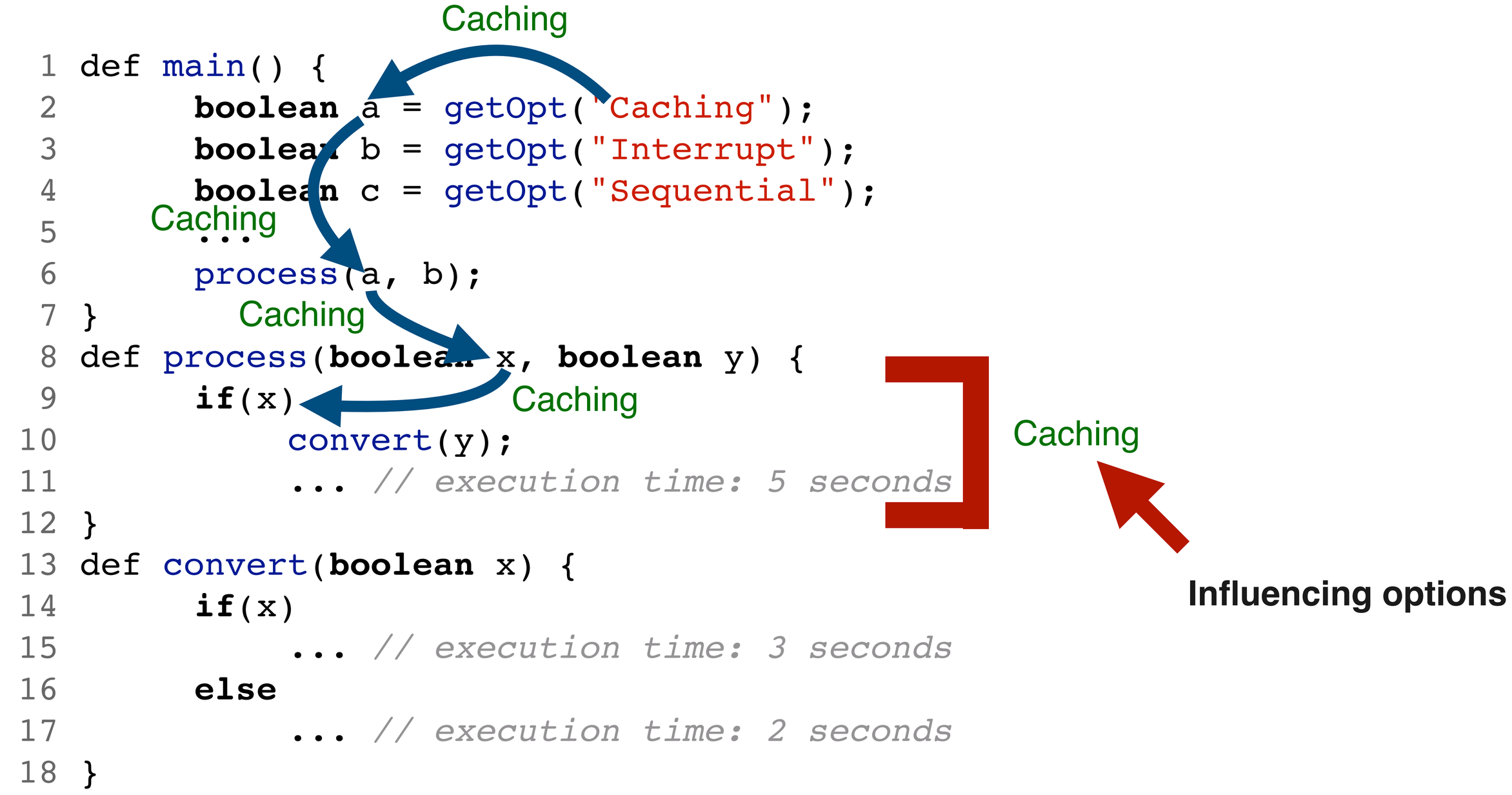
Sources



Sinks




```
1  def main() {
2        boolean a = getOpt("Caching");
3        boolean b = getOpt("Interrupt");
4        boolean c = getOpt("Sequential");
5        ...
6        process(a, b);
7  }
8  def process(boolean x, boolean y) {
9        if(x)
10           convert(y);
11           ... // execution time: 5 seconds
12 }
13 def convert(boolean x) {
14       if(x)
15           ... // execution time: 3 seconds
16       else
17           ... // execution time: 2 seconds
18 }
```




```
1  def main() {
2        boolean a = getOpt("Caching");
3        boolean b = getOpt("Interrupt");
4        boolean c = getOpt("Sequential");
5        ...
6        process(a, b);
7  }
8  def process(boolean x, boolean y) {
9        if(x)
10           convert(y);
11           ... // execution time: 5 seconds
12 }
13 def convert(boolean x) {
14       if(x)
15           ... // execution time: 3 seconds
16       else
17           ... // execution time: 2 seconds
18 }
```

```
1  def main() {
2        boolean a = getOpt("Caching");
3        boolean b = getOpt("Interrupt");
4        boolean c = getOpt("Sequential");
5        ...
6        process(a, b);
7  }
8  def process(boolean x, boolean y) {
9        if(x)
10           convert(y);
11           ... // execution time: 5 seconds
12 }
13 def convert(boolean x) {
14       if(x)
15           ... // execution time: 3 seconds
16       else
17           ... // execution time: 2 seconds
18 }
```

Caching

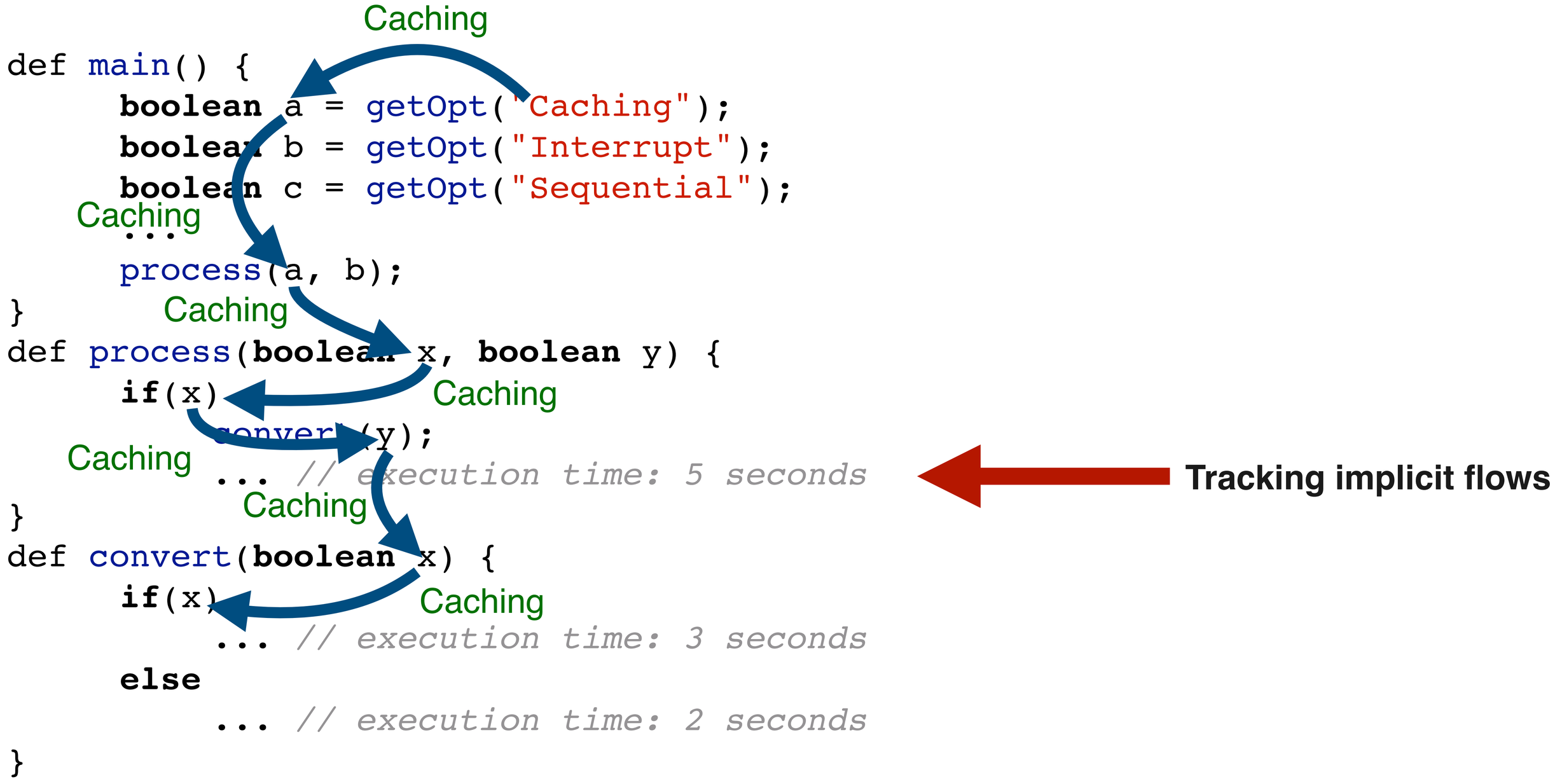
Caching

Caching

Caching

Caching	Time
TRUE	
FALSE	

Configurations to measure


Caching
1 def main() {
2 boolean a = getOpt("Caching");
3 boolean b = getOpt("Interrupt");
4 boolean c = getOpt("Sequential");
Caching
5 ...
6 process(a, b);
Caching
7 }
8 def process(boolean x, boolean y) {
Caching
9 if(x)
Caching
10 convert(y);
11 ... // execution time: 5 seconds
Caching
12 }
13 def convert(boolean x) {
Caching
14 if(x)
15 ... // execution time: 3 seconds
16 else
17 ... // execution time: 2 seconds
18 }
Tracking implicit flows

Analyze Regions

Taint Analysis

```
1  def main() {
2        boolean a = getOpt("Caching");
3        boolean b = getOpt("Interrupt");
4        boolean c = getOpt("Sequential");
5        ...
6        process(a, b);
7  }
8  def process(boolean x, boolean y) {
9        if(x)
10           convert(y);
11       ... // execution time: 5 seconds
12 }
13 def convert(boolean x) {
14       if(x)
15           ... // execution time: 3 seconds
16       else
17           ... // execution time: 2 seconds
18 }
```

Caching
Caching
Caching
Caching
Caching
Caching

Measure Performance

Compression

Caching	Interrupt	Sequential
FALSE	FALSE	FALSE
TRUE	TRUE	TRUE

```
1  def main() {
2        boolean a = getOpt("Caching");
3        boolean b = getOpt("Interrupt");
4        boolean c = getOpt("Sequential");
5        ...
6        if(a) // variable depends on Caching
7            ... // execution time: 1 second
8        if(b) // variable depends on Interrupt
9            ... // execution time: 2 seconds
10       if(c) // variable depends on Sequential
11           ... // execution time: 3 seconds
12 }
```

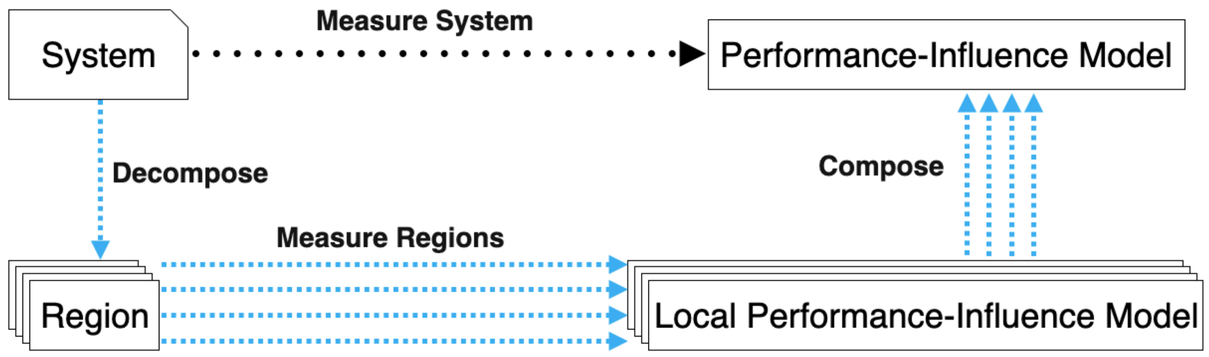
Caching	Time
TRUE	1
FALSE	0

Interrupt	Time
TRUE	2
FALSE	0

Sequential	Time
TRUE	3
FALSE	0

Build Model

Compositionality



ConfigCrusher	Comprex
ASEJ'20	Under submission
Static Taint Analysis	(Iterative) Dynamic Taint Analysis
Regions: Control-flow statements	Regions: Methods

Both prototypes efficiently build accurate and interpretable models

Dynamic taint analysis scales to larger systems

Method-level granularity does not sacrifices compression potential

Evaluation

ConfigCrusher

ASEJ'20

Comprex

Under submission

32 combinations of sampling and machine learning approaches

13 open-source Java systems

Evaluate cost, accuracy, interpretability

Subject systems

System	# SLOC	# Options	# Configurations
Pngtastic Counter	1250	5	32
Pngtastic Optimizer	2553	5	32
Elevator	575	6	20
Grep	2152	7	128
Kanzi	20K	7	128
Email	696	9	40
Prevayler	1328	9	512
Sort	2163	12	4096
H2	142K	16	65K
Berkeley DB	164K	16	65K
Apache Lucene	396K	17	131K
Density Converter	49K	22	4.9M

Density Converter

22 options, 4.9 million configurations

Metric	200 Random & Stepwise Linear Regression	200 Random & Random Forest	Comprex

Density Converter

22 options, 4.9 million configurations

Metric	200 Random & Stepwise Linear Regression	200 Random & Random Forest	Comprex
# Configurations	200	200	**88**
Measuring time	40.4 minutes	40.4 minutes	**16.6 minutes**

Density Converter

22 options, 4.9 million configurations

Metric	200 Random & Stepwise Linear Regression	200 Random & Random Forest	Comprex
# Configurations	200	200	**88**
Measuring time	40.4 minutes	40.4 minutes	**16.6 minutes**
Learning/Analysis Time	1.6 minutes	**0.2 seconds**	8.5 minutes

Density Converter

22 options, 4.9 million configurations

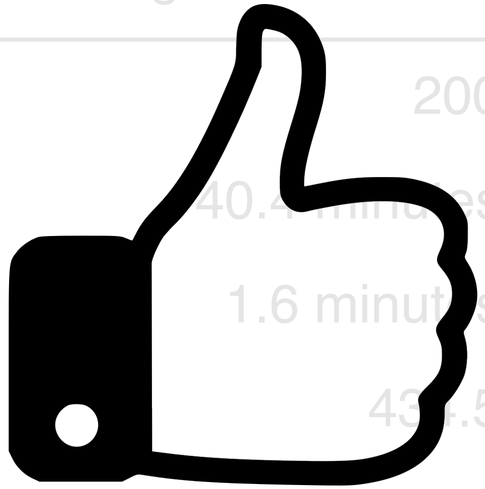
Metric	200 Random & Stepwise Linear Regression	200 Random & Random Forest	Comprex
# Configurations	200	200	**88**
Measuring time	40.4 minutes	40.4 minutes	**16.6 minutes**
Learning/Analysis Time	1.6 minutes	**0.2 seconds**	8.5 minutes
MAPE (lower is better)	434.5	**5.5**	**9.4**

Density Converter

22 options, 4.9 million configurations

Efficient and accurate predictions

Metric	200 Random & Stepwise Linear Regression	200 Random & Random Forest	Comprex
# Configurations	200	200	**88**
Measuring time	40.4 minutes	40.4 minutes	**16.6 minutes**
Learning/Analysis Time	1.6 minutes	**0.2 seconds**	8.5 minutes
MAPE (lower is better)	43.5	**5.5**	**9.4**



Density Converter

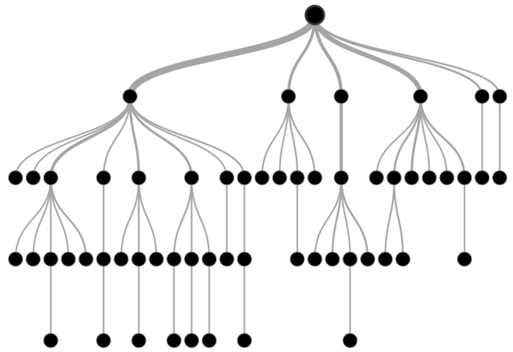
22 options, 4.9 million configurations

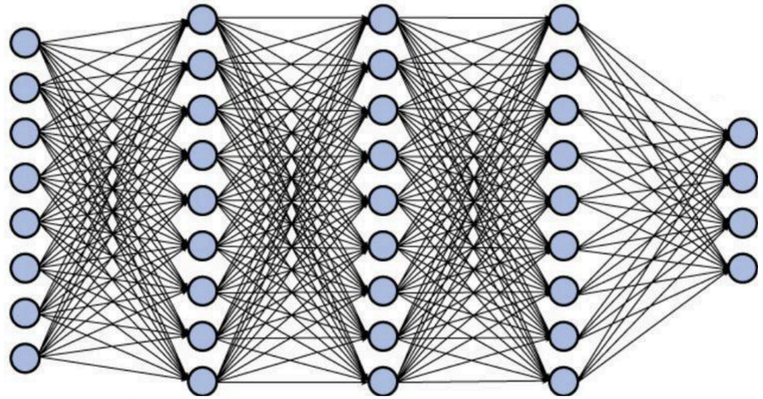
Metric	200 Random & Stepwise Linear Regression	200 Random & Random Forest	Comprex
# Configurations	200	200	**88**
Measuring time	40.4 minutes	40.4 minutes	**16.6 minutes**
Learning/Analysis Time	1.6 minutes	**0.2 seconds**	8.5 minutes
MAPE (lower is better)	434.5	**5.5**	**9.4**
Interpretability	**Easy**	Difficult	**Easy**

Inherently interpretable

```
T = 25
    + 3·Logging
    - 5·Indexing
    + 9·Compression·Encryption
```

Not inherently interpretable





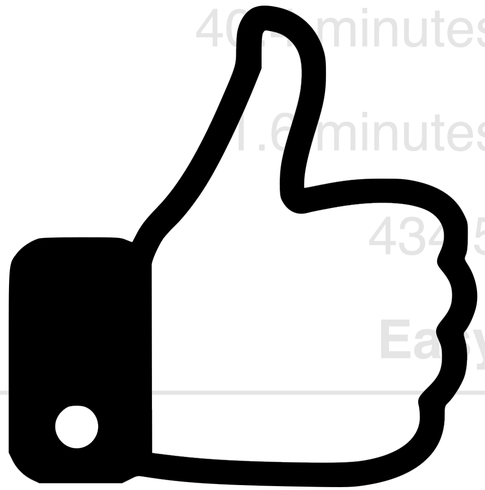
Density Converter

22 options, 4.9 million configurations

Efficient and accurate predictions

Understanding and debugging

Metric	200 Random & Stepwise Linear Regression	200 Random & Random Forest	Comprex
# Configurations	200	200	**88**
Measuring time	40.4 minutes	40.4 minutes	**16.6 minutes**
Learning/Analysis Time	1.6 minutes	**0.2 seconds**	8.5 minutes
MAPE (lower is better)	43.5	**5.5**	**9.4**
Interpretability	**Easy**	Difficult	**Easy**



Density Converter

Efficiently build accurate performance-influence models

Metric	200 Random & Stepwise Linear Regression	200 Random & Random Forest	Comprex
# Configurations	200	200	**88**
Measuring time	40.4 minutes	40.4 minutes	**16.6 minutes**
Learning/Analysis Time	1.6 minutes	**0.2 seconds**	8.5 minutes
MAPE (lower is better)	434.5	**5.5**	**9.4**
Interpretability	**Easy**	Difficult	**Easy**

Density Converter

Efficiently build accurate performance-influence models

Metric	200 Random & Stepwise Linear Regression	200 Random & Random Forest	Comprex
# Configurations	200	200	**88**
Measuring time	40.4 minutes	40.4 minutes	**16.6 minutes**
Learning and Tree	1.6 minutes	**0.2 seconds**	8.5 minutes
MAPE (lower is better)	434.5	**5.5**	**9.4**
Interpretability	**Easy**	Difficult	**Easy**

Models are interpretable and can be mapped to regions

Density Converter

Proposed Work

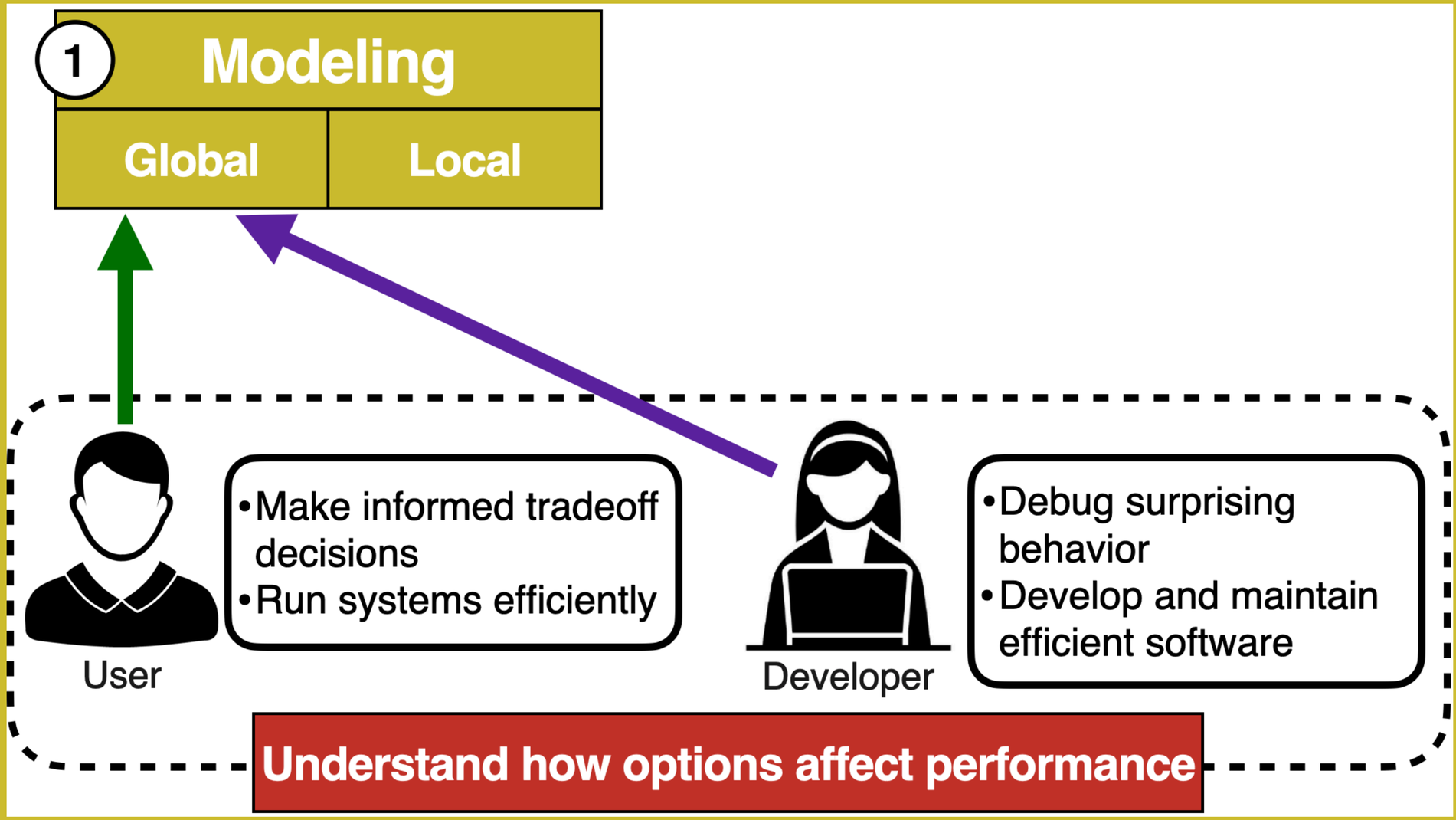
22 options, 4.9 million configurations

Analyze all subject systems with both prototypes

Compare both prototypes to all state of the art approaches

Metric	200 Random & Stepwise Linear Regression	200 Random & Random Forest	Comprex
# Configurations	200	200	**88**
Measuring time	40.4 minutes	40.4 minutes	**16.6 minutes**
Learning/Analysis Time	1.6 minutes	**0.2 seconds**	8.5 minutes
MAPE (lower is better)	434.5	**5.5**	**9.4**
Interpretability	**Easy**	Difficult	**Easy**

White-box Performance Modeling of Configurable Systems



Thesis Statement

White-box analysis of how options influence the performance of code-level structures in configurable systems:

(1) helps to **efficiently build accurate and interpretable global and local performance-influence models**

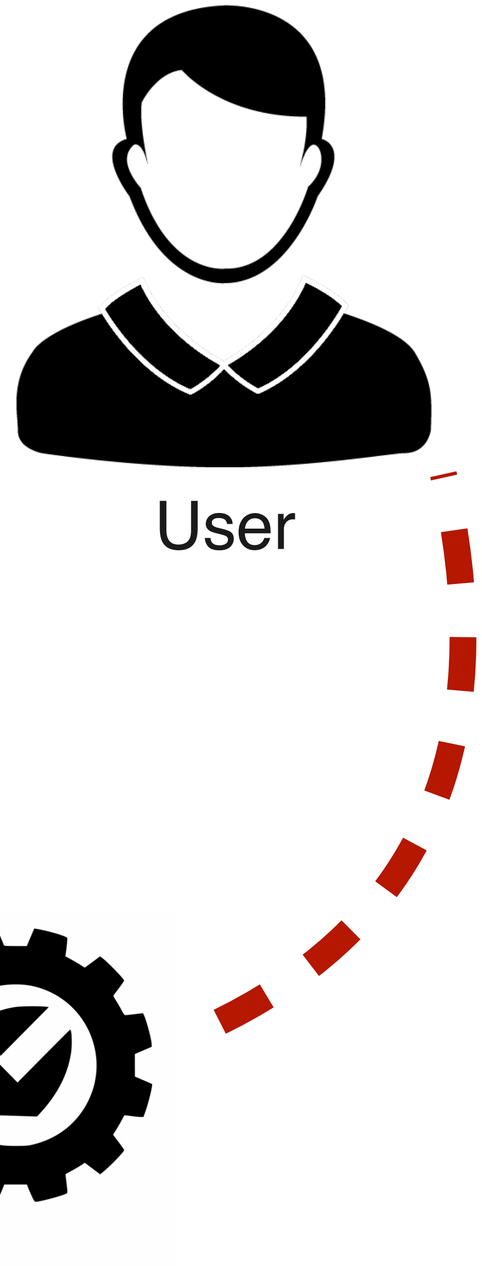
(2) guides developers to **inspect, understand, and debug configuration-related performance behaviors**.

```
T = 25
    + 3·Logging
    - 5·Indexing
    + 9·Compression·Encryption
```

User

System

```
T = 25
    + 3·Logging
    - 5·Indexing
    + 9·Compression·Encryption
```



System

1 **Modeling**

Global | **Local**

User

- Make informed tradeoff decisions
- Run systems efficiently

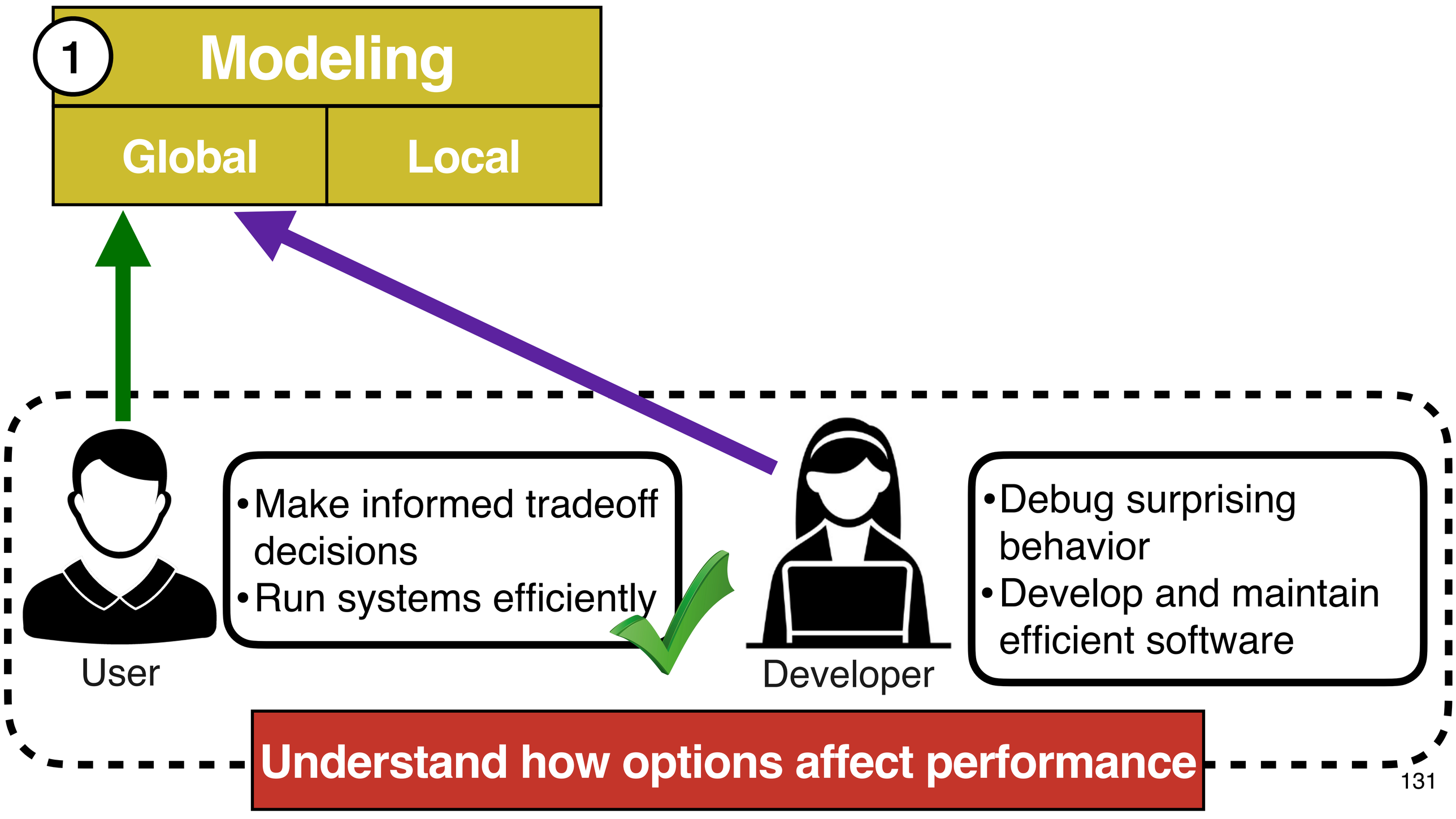
Developer

- Debug surprising behavior
- Develop and maintain efficient software

Understand how options affect performance

```
T = 25
    + 3·Logging
    - 5·Indexing
    + 9·Compression·Encryption
```

Developer


1
Modeling
Global
Local
User
• Make informed tradeoff decisions
• Run systems efficiently
Developer
• Debug surprising behavior
• Develop and maintain efficient software
Understand how options affect performance

1 **Modeling**

Global	Local

But

User

- Make [illegible] meo [illegible] eoff deci[illegible]
- [illegible]stems en[illegible]

Developer

- Debug surprising behavior
- [illegible]evelop and maintain efficient software

Understand how options affect performance

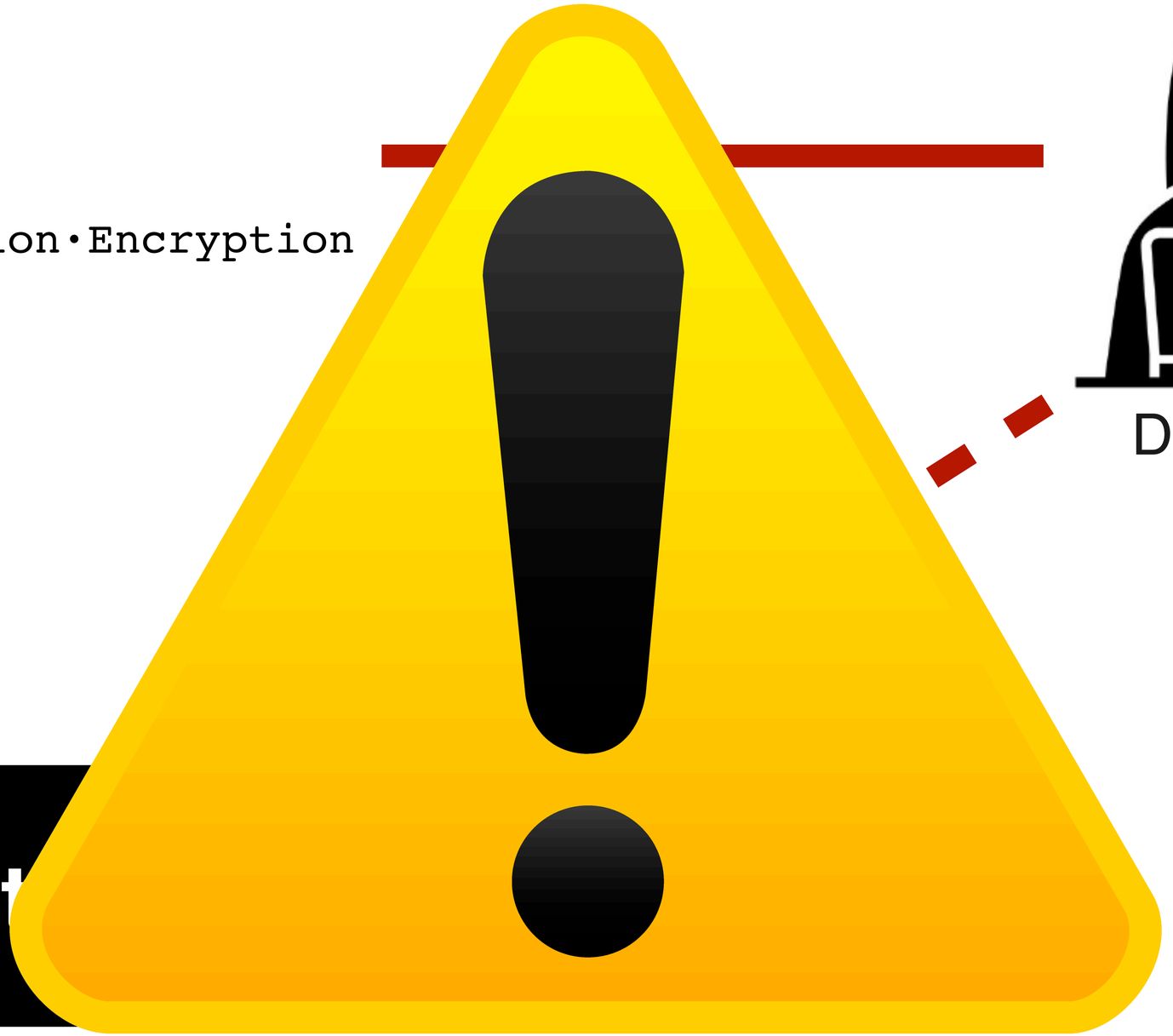
```
T = 25
      + 3·Logging
      - 5·Indexing
      + 9·Compression·Encryption
```

Developer

System

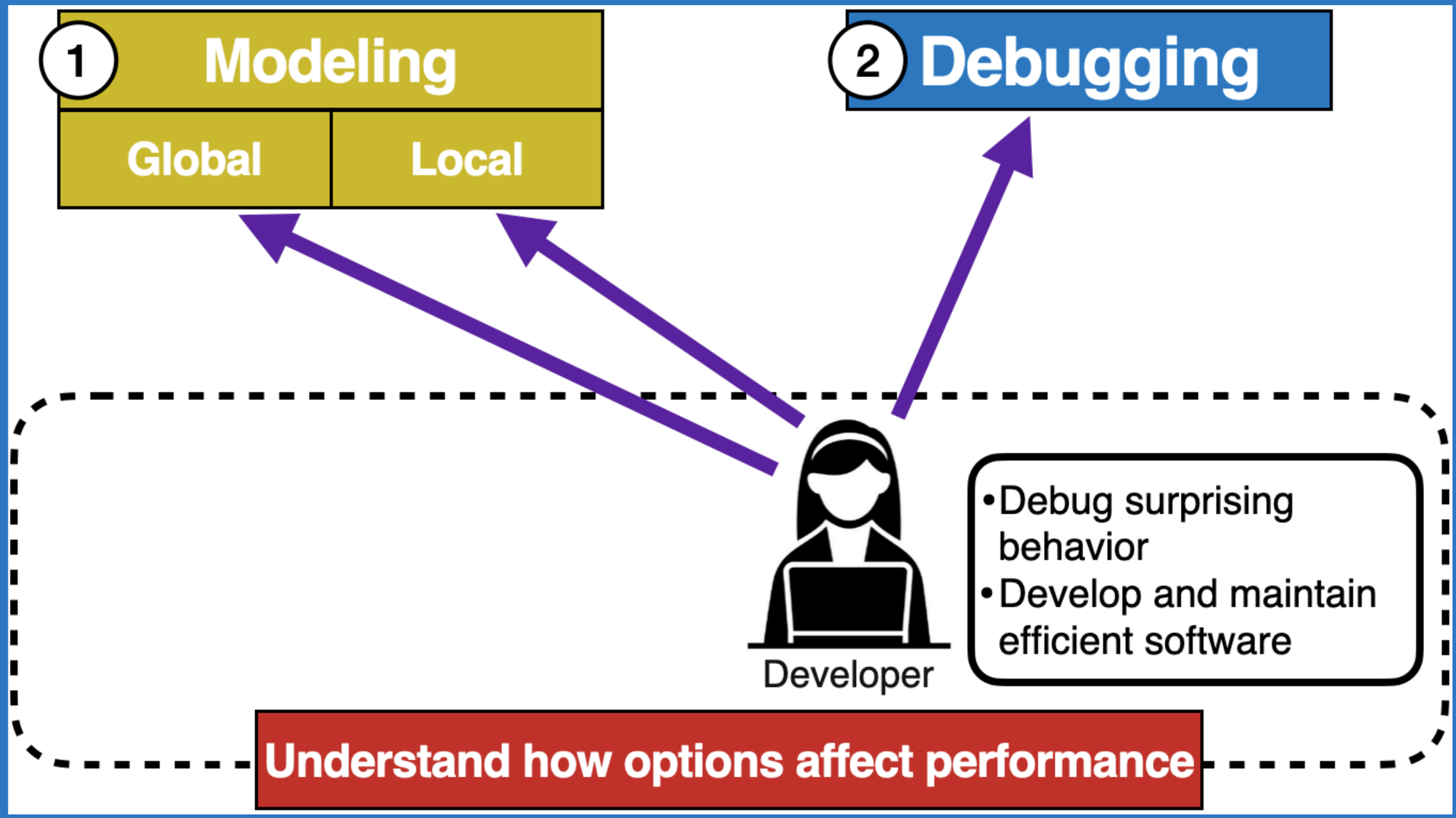
```
T = 25
    + 3·Logging
    - 5·Indexing
    + 9·Compression·Encryption
```

Developer



Syst

White-box Performance Debugging in Configurable Systems

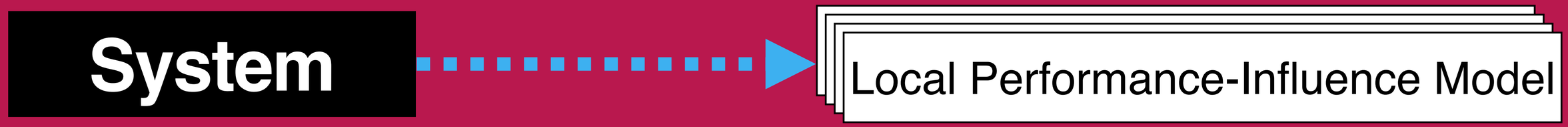


```
T = 25
    + 3·Logging
    - 5·Indexing
    + 9·Compression·Encryption
```

Developer

System

Recall Compositionality



Local performance-influence models explain how options and their interactions affect the performance of regions

Exploratory Study of Local Performance-Influence Models

Goal:

- **Investigate the usefulness of local performance-influence models**
- **Understand how options affect the performance of a system in the implementation**

Exploratory Study of Local Performance-Influence Models

Symptoms vs causes

Goal:
Investigate the usefulness of local performance-influence models
Understand how options affect the performance of a system in the implementation

Symptom

```
1  def main() {
2        boolean a = getOpt(“Caching");
3        boolean z;
4        if(a)
5             z = true;
6        else
7             z = false;
8        process(z)
9  }
10 def process(boolean x) {
11       if(x)
12            ... // 3 seconds
13       else
14            ... // 2 seconds
15 }
```

$$T_{process} = 2 + 1 \cdot Caching$$

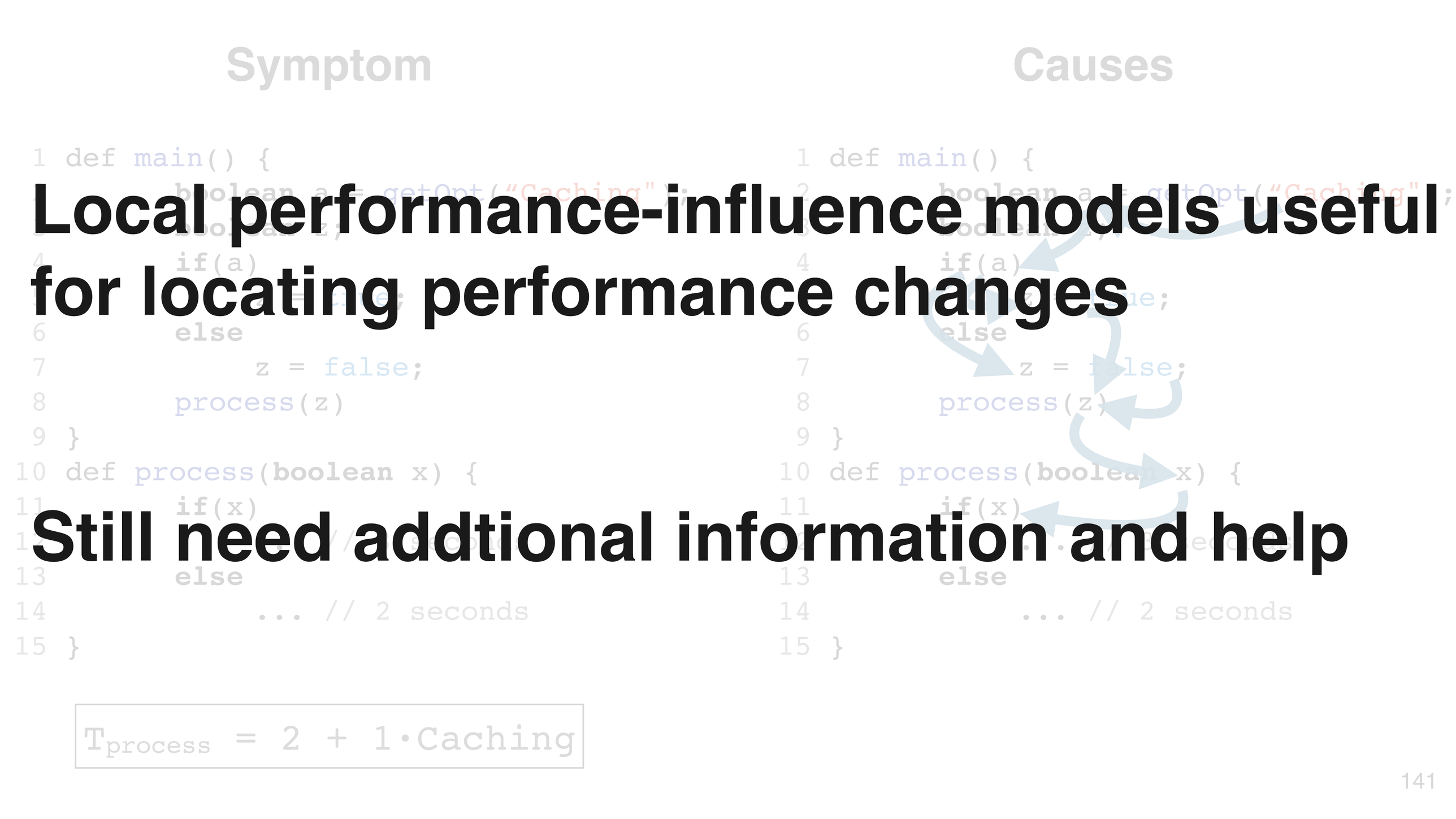
Causes

```
1  def main() {
2        boolean a = getOpt(“Caching");
3        boolean z;
4        if(a)
5             z = true;
6        else
7             z = false;
8        process(z)
9  }
10 def process(boolean x) {
11       if(x)
12            ... // 3 seconds
13       else
14            ... // 2 seconds
15 }
```

Symptom

Causes

```
1 def main() {
2     boolean a = getOpt("Caching");
3     boolean z;
4     if(a)
5         z = true;
6     else
7         z = false;
8     process(z)
9 }
10 def process(boolean x) {
11    if(x)
12        ... // 2 seconds
13    else
14        ... // 2 seconds
15 }
```



Local performance-influence models useful for locating performance changes

Still need addtional information and help

$$T_{process} = 2 + 1 \cdot Caching$$

Design Tool Support

Goal: Design and develop debugging tool support

Two-Part Exploratory Study

Research Question: What is the process and information needs when debugging the performance of configurable systems?

Begin debugging performance

Debug Performance of Hotspot Regions

Study 1.1

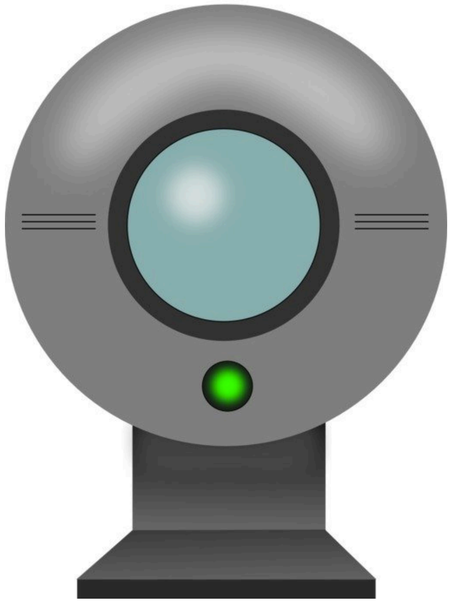
Goal: Are global and local performance-influence models helpful?

Study 1.2

Later in the debugging process

Identified options and locations of interest

Goal: Guide design of new tool support



Conducted studies with 14 researchers/developers working in academia

Expect to conduct studies with 5 developers working in industry

Preliminary Results - Study 1.1 - Process

1. Which options affect performance the most?

Hypothesis: Global performance-influence models

Conducted studies with 14 researchers/developers working in academia

Expect to conduct studies with 5 developers working in industry

```
T = 25
    + 3·Logging
    - 5·Indexing
    + 9·Compression·Encryption
```

Preliminary Results - Study 1.1 - Process

2. Where do options affect performance the most?

Hypothesis: Local performance-influence models

Conducted studies with 14 researchers/developers working in academia

$$T_{main} = 2 \qquad T_{process} = 2 \cdot Compression \cdot Encryption$$

Expect to conduct studies with 5 developers working in industry

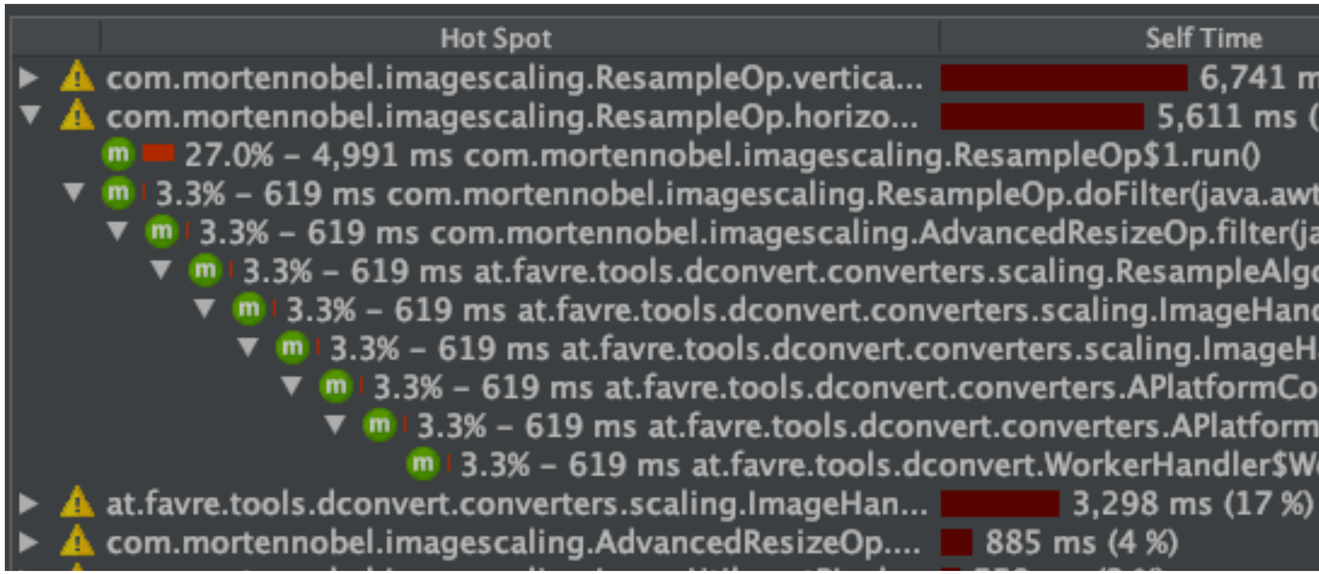
$$T_{convert} = 7 \cdot Compression \cdot Encryption$$

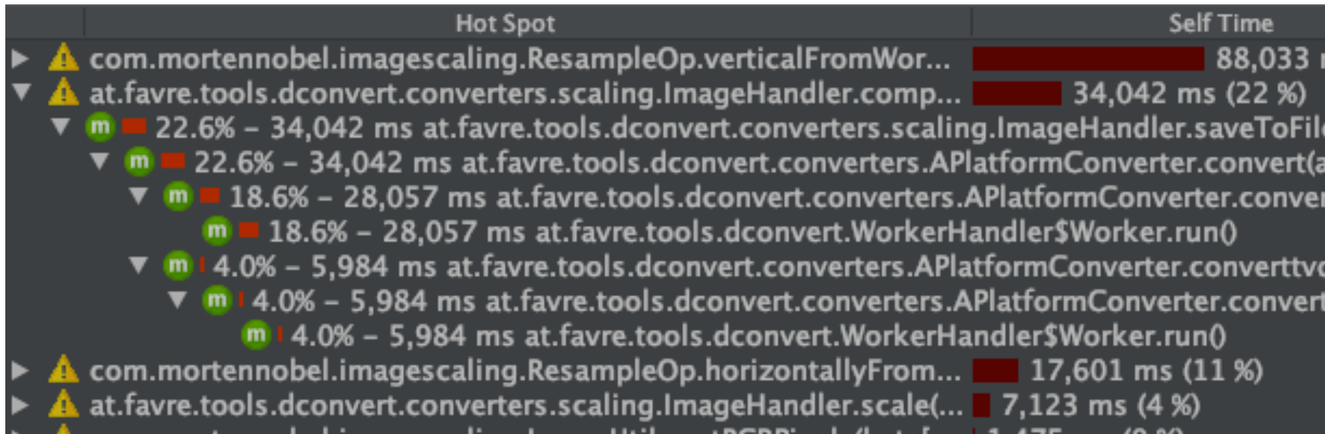
Preliminary Results - Study 1.2 - Information needs

Trace options

```
1  def main() {
2        boolean a = getOpt(“Caching");
3        boolean z;
4        if(a)
5             z = true;
6        else
7             z = false;
8        process(z)
9  }
10 def process(boolean x) {
11       if(x)
12            ... // 3 seconds
13       else
14            ... // 2 seconds
15 }
```

Compare performance profiles





Proposed Work

Trace options

Compare performance profiles

```
1  def main() {
2          boolean z;
3          boolean a;
4          if(a)
5                  z = true;
6          else
7                  z = false;
8          process(z)
9  }
10 def process(boolean x) {
11         if(x)
12                 ... // 3 seconds
13         else
14                 ... // 2 seconds
15 }
```

Conduct studies with 5 developers working in industry

Quantitatively and qualitatively analyze process and information needs

Design Tool Support

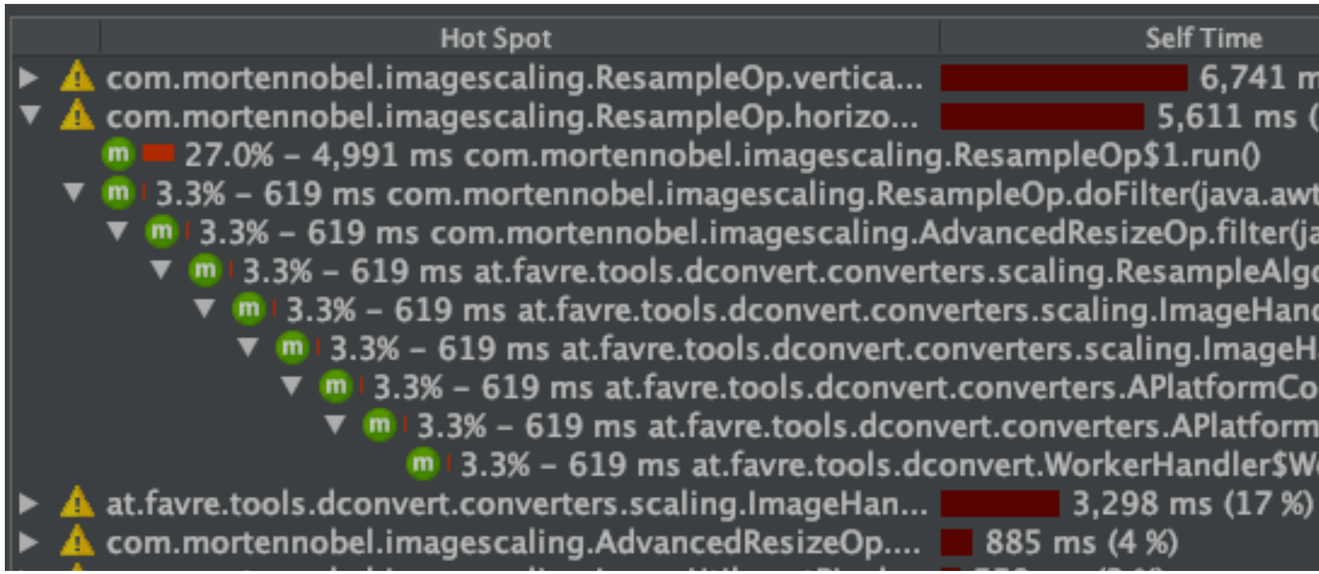
Goal: Design and develop debugging tool support

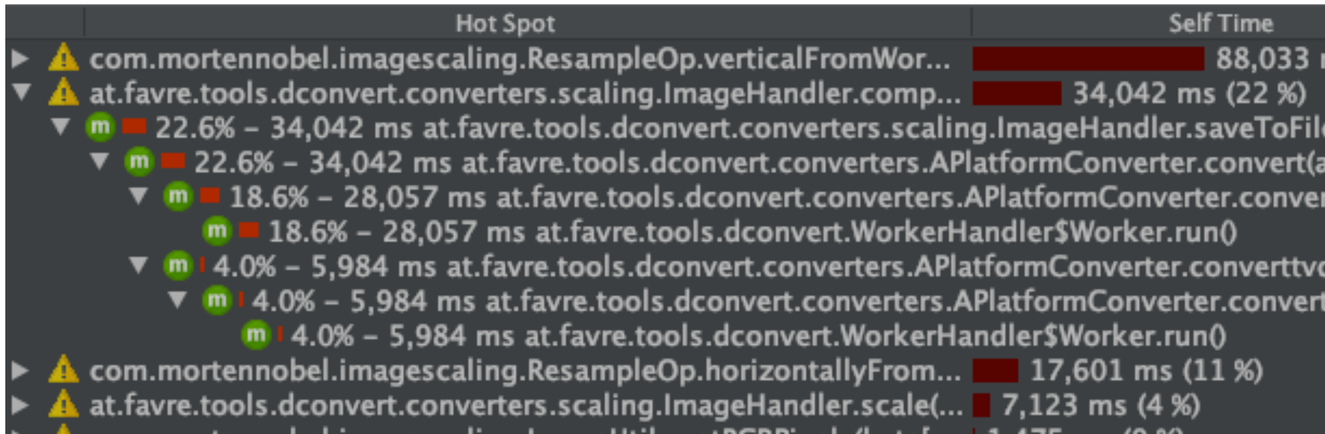
Examples of Tool Support

Trace options

```
1  def main() {
2          boolean a = getOpt(“Caching");
3          boolean z;
4          if(a)
5              z = true;
6          else
7              z = false;
8          process(z)
9  }
10 def process(boolean x) {
11         if(x)
12             ... // 3 seconds
13         else
14             ... // 2 seconds
15 }
```

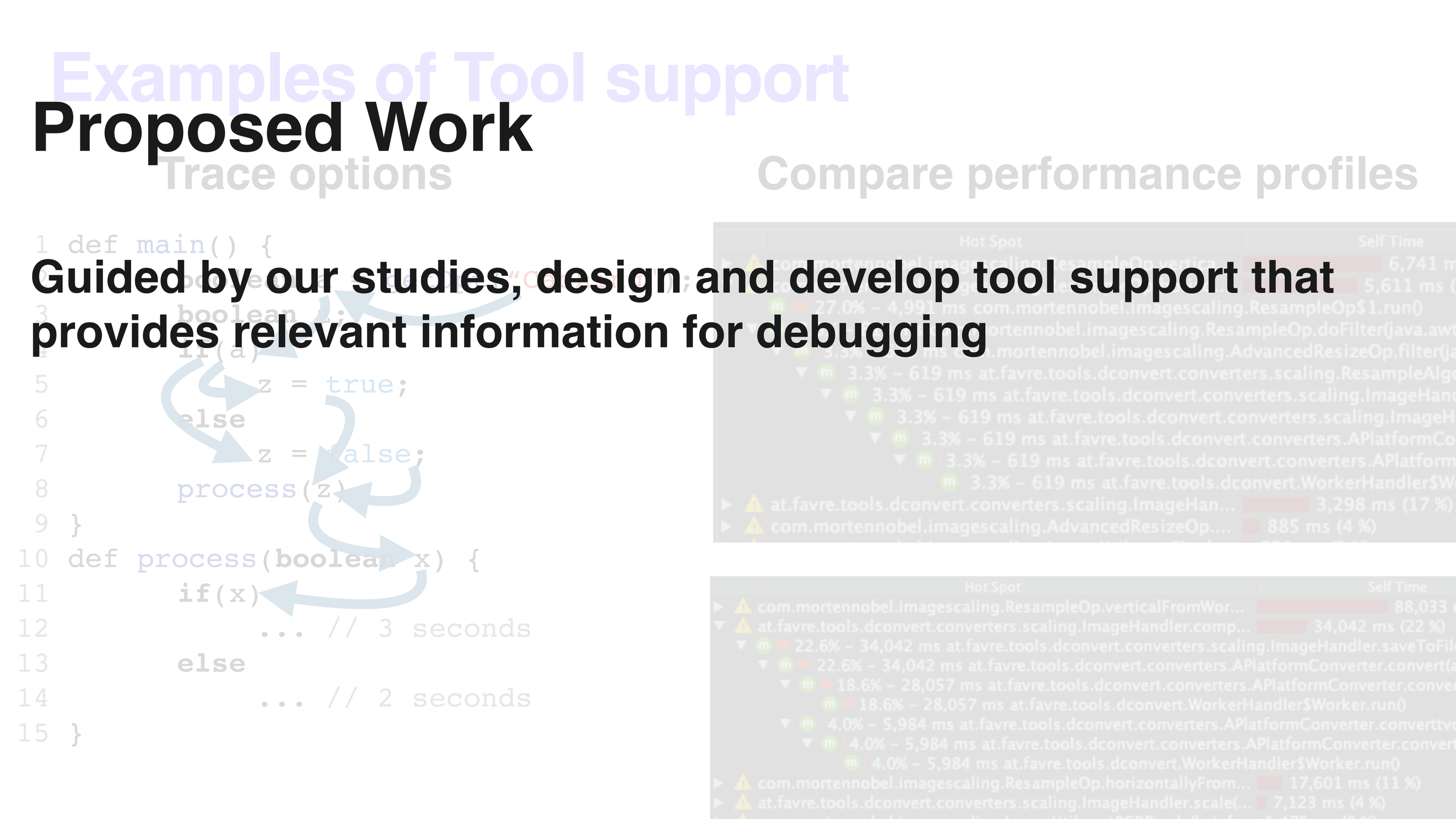
Compare performance profiles





Proposed Work

Guided by our studies, design and develop tool support that provides relevant information for debugging



Validation Study

Study 2

Research Questions: To what extend are

(1) Global performance-influence models

(2) Local performance-influence models

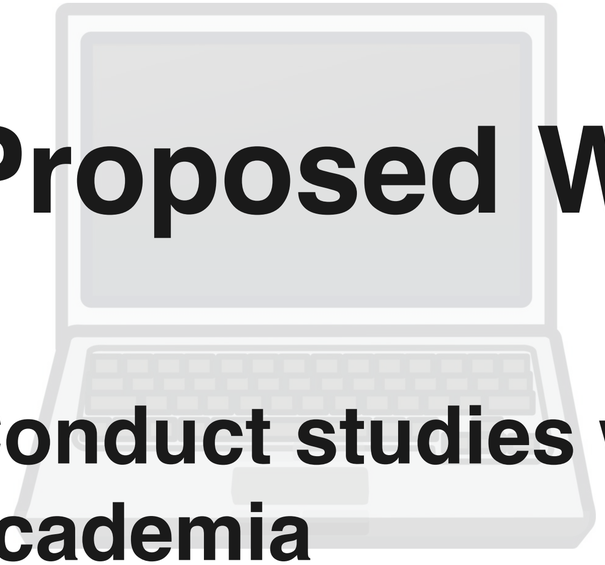
(3) Additional tool support

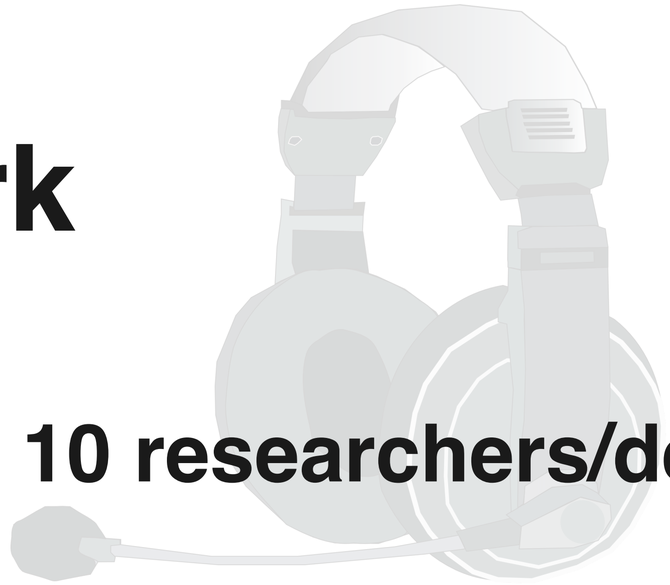
useful to debug the performance of configurable systems?

Expect to conduct studies with 10 researchers/developers working in academia

No control group: Participants cannot debug without tool support (Study 1.1 and Study 1.2)

Proposed Work





Conduct studies with 10 researchers/developers working in academia

Expect to conduct studies with 10 researchers/developers working in academia

No control group: Participants cannot debug without tool support (Study 1.1 and Study 1.2)

Optional Validation Study in the Field

Optional Study 3

Goal: Validate the usefulness of our tools in real bug reports and real systems

Optional Validation Study in the Field

Option 1: Deploy tools in the field

Option 2: Collaborate with developers

Goal: Validate the usefulness of our tools in real bug reports and real systems

Option 3: Case studies

Option 1: Deploy tools in the field

Option 2: Collaborate with developers

Goal: Validate the usefulness of our tools in real bug reports and real systems

Option 3: Case studies

Optional Validation Study in the Field

Option 1: Deploy tools in the field

Option 2: Collaborate with developers

Goal: Validate the usefulness of our tools in real bug reports and real systems

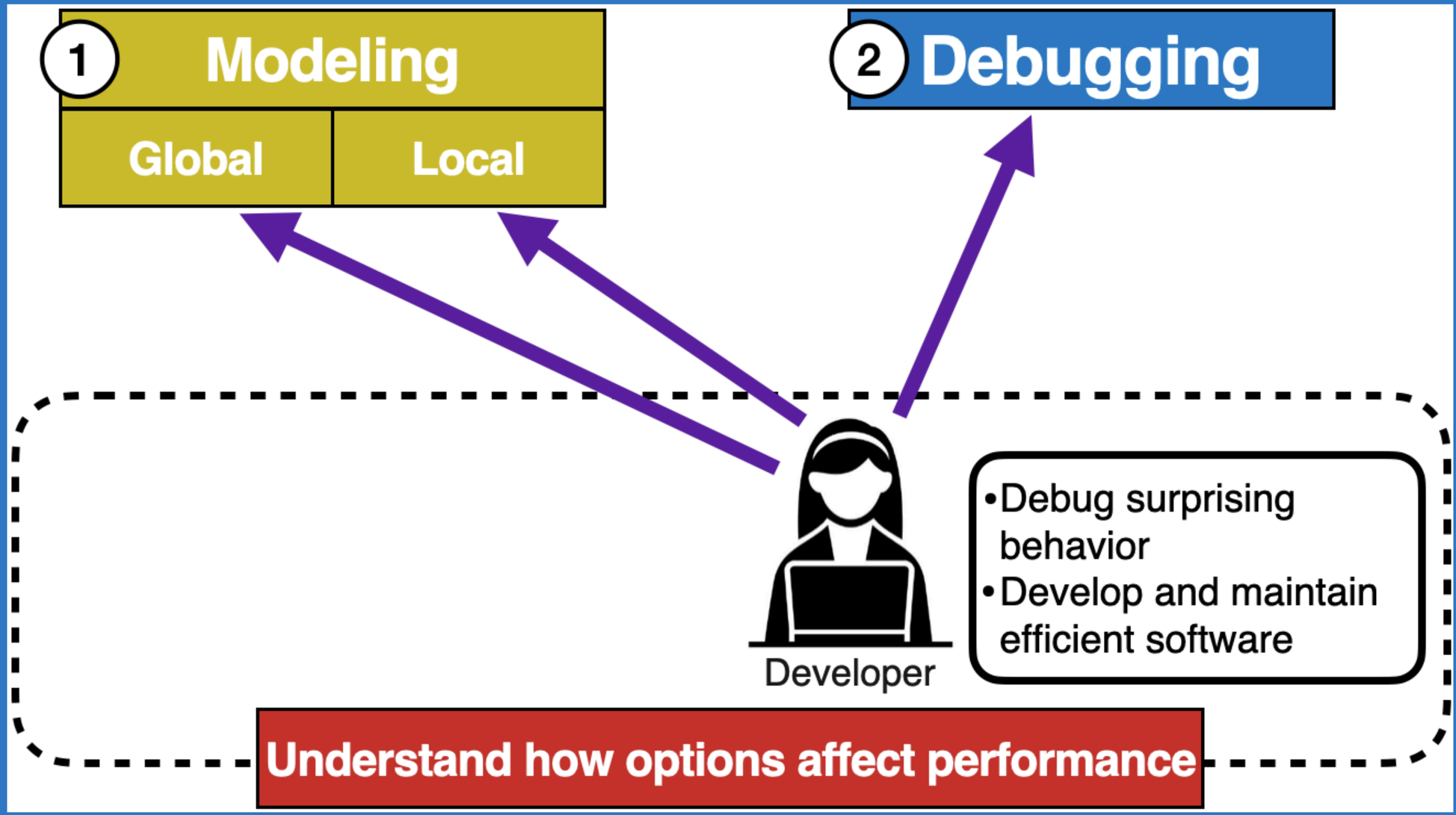
Option 3: Case studies

Optional Validation Study in the Field

Optional Study 3

Goal: Validate the usefulness of our tools in real bug reports and real systems

White-box Performance Debugging in Configurable Systems



Thesis Statement

White-box analysis of how options influence the performance of code-level structures in configurable systems:

(1) helps to **efficiently build accurate and interpretable global and local performance-influence models**

(2) guides developers to **inspect, understand, and debug configuration-related performance behaviors**.

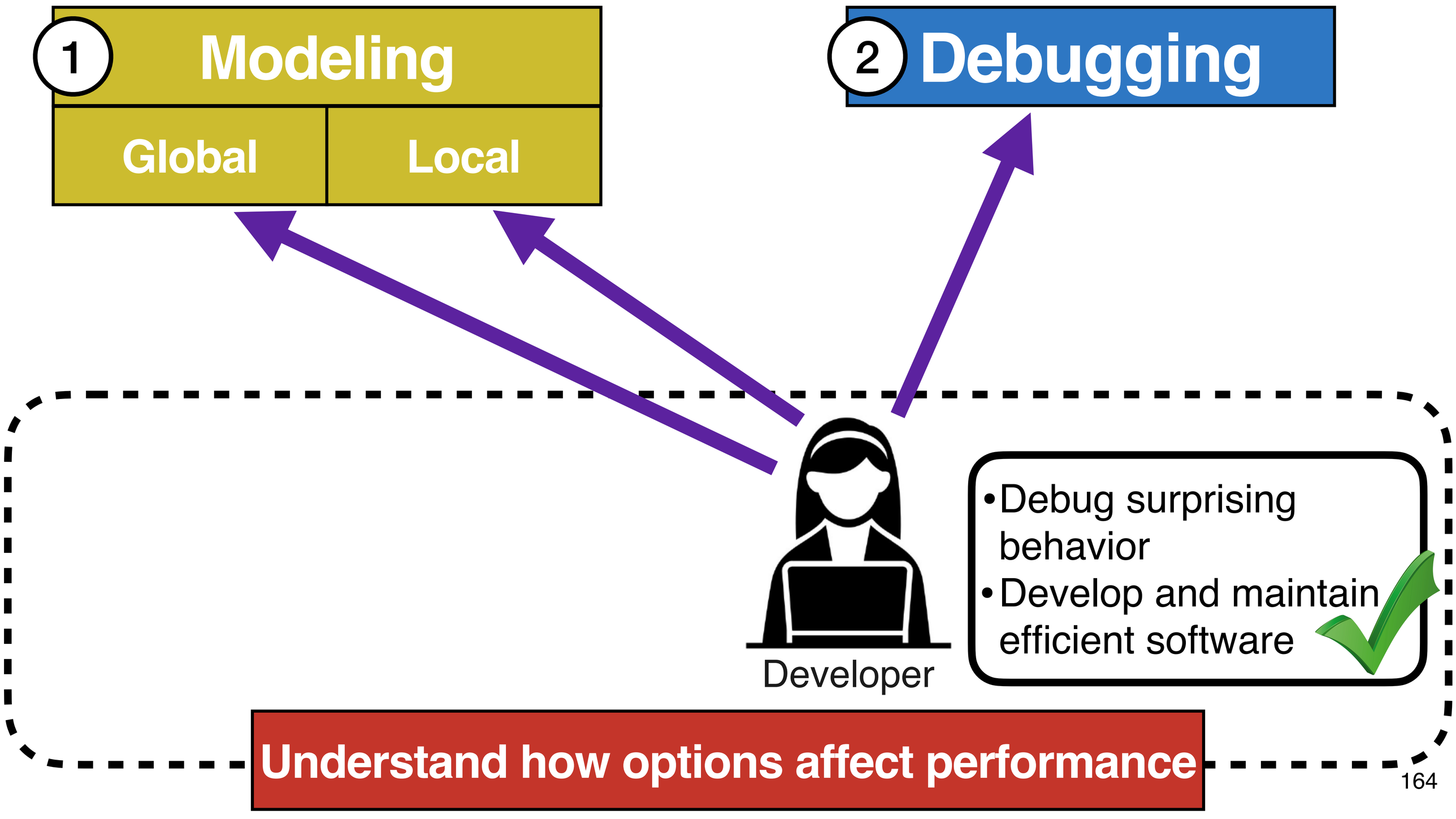
$$T = 25 + 3 \cdot \text{Logging} - 5 \cdot \text{Indexing} + 9 \cdot \text{Compression} \cdot \text{Encryption}$$

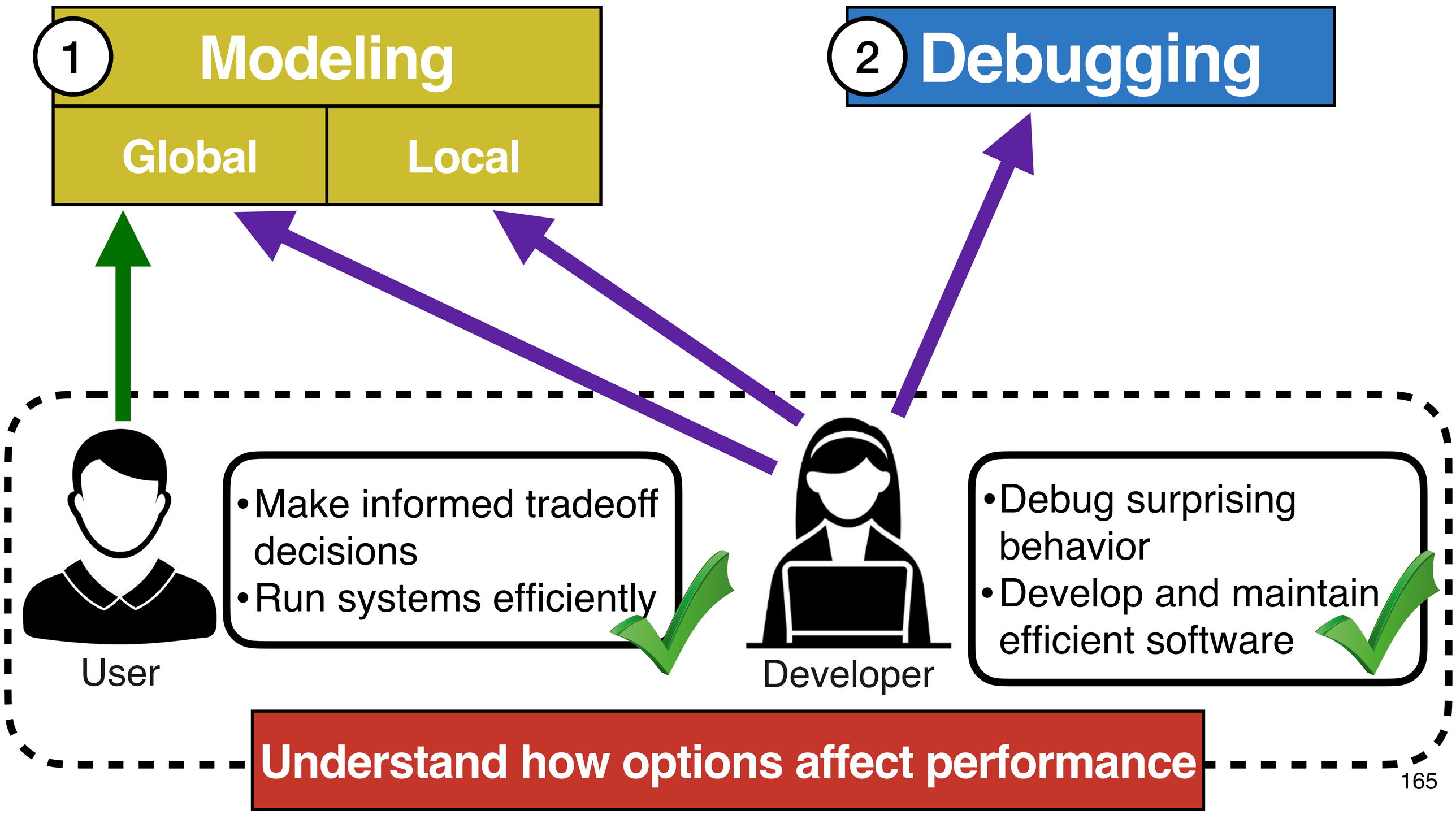
Developer

System

$$T_{compress} = 5 \cdot \text{Indexing}$$

$$T_{process} = 9 \cdot \text{Compression} \cdot \text{Encryption}$$


1 Modeling
Global
Local
2 Debugging
Developer
•Debug surprising behavior
•Develop and maintain efficient software
Understand how options affect performance


1
Modeling
Global
Local
2
Debugging
User
• Make informed tradeoff decisions
• Run systems efficiently
Developer
• Debug surprising behavior
• Develop and maintain efficient software
Understand how options affect performance

1 Modeling

Global | Local

2 Debugging

Research Plan

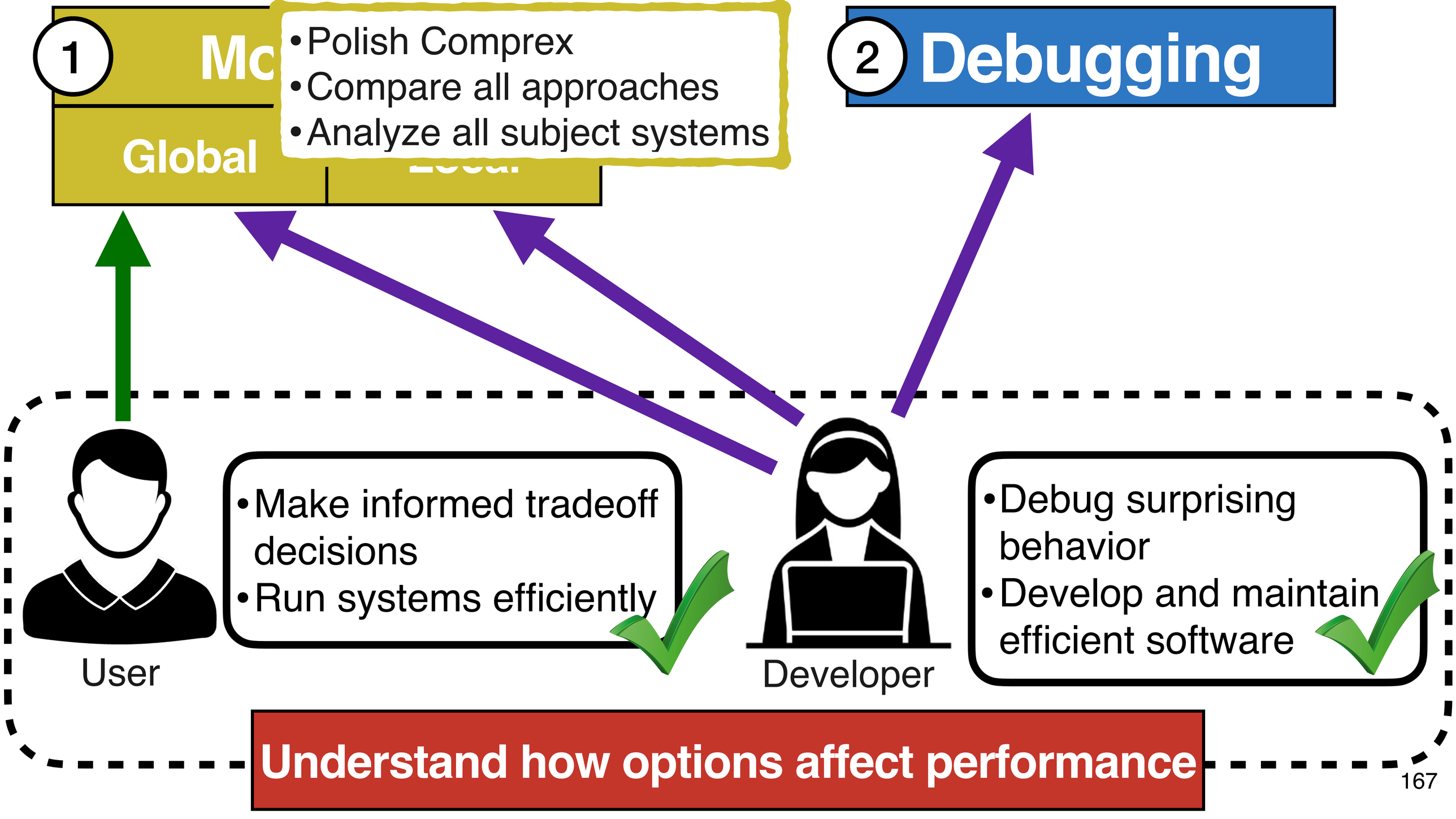
User

- Make informed trade-off decisions
- Run systems efficiently

Developer

- Debug surprising behavior
- Develop and maintain efficient software

Understand how options affect performance


1
Mc
Global
•Polish Comprex
•Compare all approaches
•Analyze all subject systems
2 Debugging
•Make informed tradeoff decisions
•Run systems efficiently
User
Developer
•Debug surprising behavior
•Develop and maintain efficient software
Understand how options affect performance

1 Mc

Global

- Polish Comprex
- Compare all approaches
- Analyze all subject systems

2 Debugging

- Finish Studies 1.1 and 1.2
- Analyze Studies
- Implement new tool support
- Study 2
- Optional Study 3

User

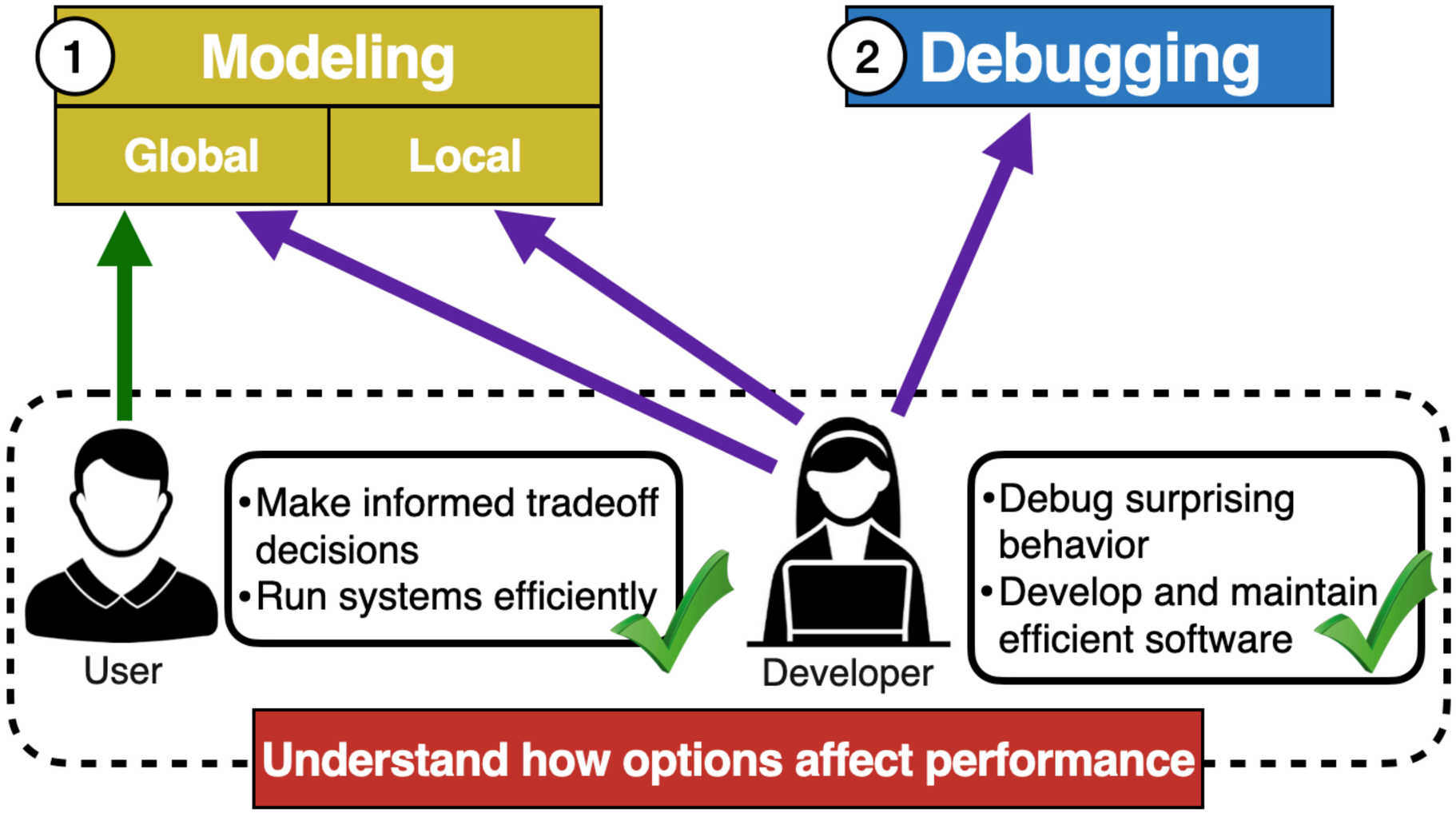
- Make informed tradeoff decisions
- Run systems efficiently

Developer

- Debug surprising behavior
- Develop and maintain efficient software

Understand how options affect performance

White-box Analysis for Modeling and Debugging the Performance of Configurable Systems



Extra slides

Sparse Linear Models

	Apache Lucene	H2	Berkeley DB	Density Converter
Terms	27	33	77	47
Regions	24	5	17	9

Modeling	1.5 months
Polish Comprex writing	2 weeks
Compare prototypes with all approaches	2 weeks
Analyze all subject systems with both prototypes	2 weeks

Debugging	11 months
Study 1.1 and Study 1.2 (Exploration)	2 months
Design and implement new tool support	6 months
Study 2 (Validation)	2 months
Optional Study 3 (Validation in the field)	2 months
Writing	1 month

Thesis writing and defense	2 months

Implicit flows

Region Analysis

Preliminary Results - Study 1.1 - Process