Characterizing the Energy Efficiency of Java's Thread-Safe Collections in a Multi-Core Environment


Gustavo Pinto


Kenan Liu


Fernando Castor


David Liu

INSTITUTO FEDERAL PARÁ

BINGHAMTON UNIVERSITY STATE UNIVERSITY OF NEW YORK

Centro de Informática UFPE

VIRTUS IMPAVIDA

BINGHAMTON UNIVERSITY STATE UNIVERSITY OF NEW YORK

Motivation (1/3)

- First, energy consumption is a concern for unwired devices and also for data centers

- Second, there is a large body of work in hardware/ architecture, OS, runtime systems
- However, little is known about the application level

Motivation (2/3)

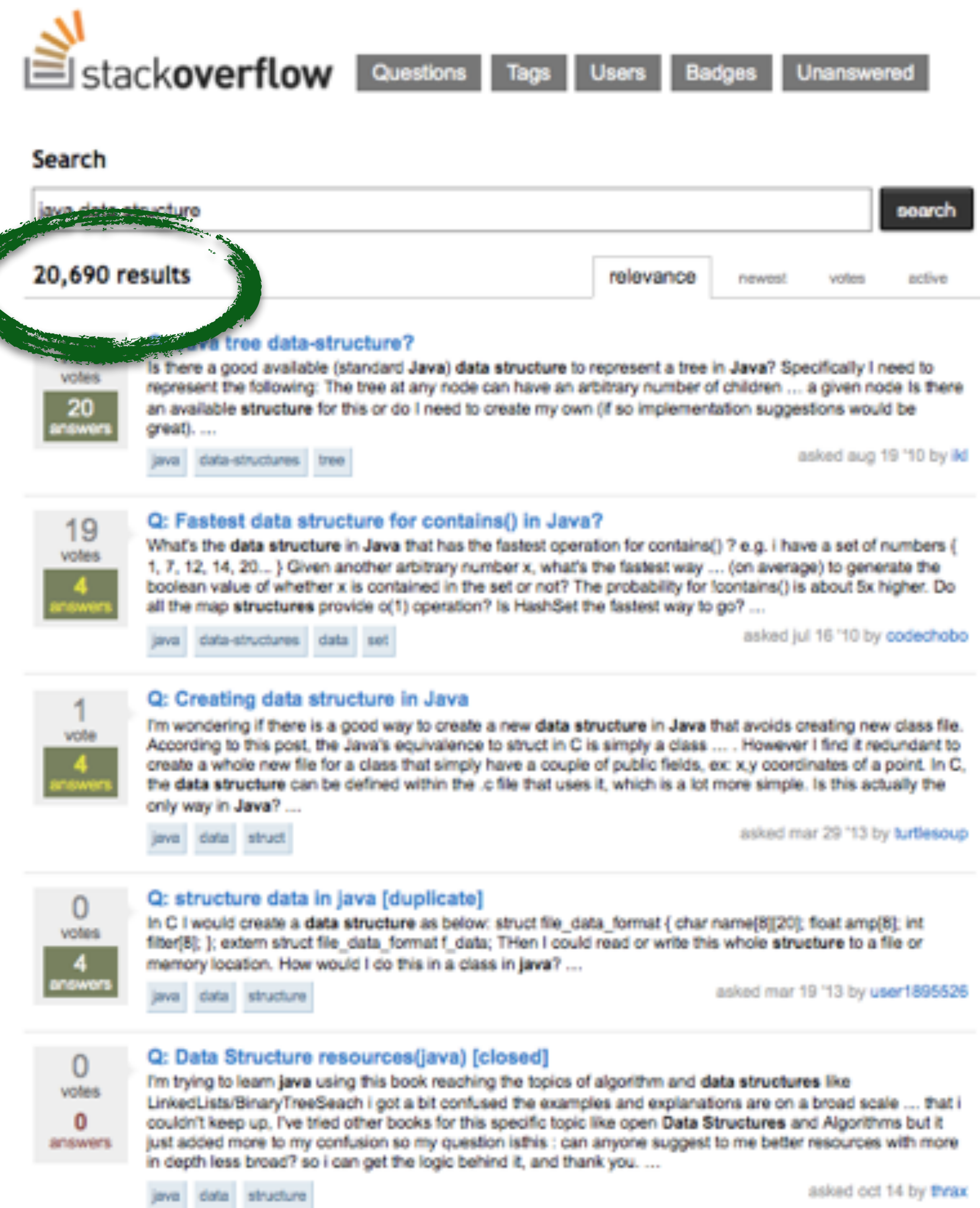
- First, multicore CPUs are ubiquitous
- Second, more cores used → more power dissipated
- However, little is known about the energy-efficiency of multicore programs

Motivation (3/3)

- Data structures are the fundamentals of computer programming

Bad programmers worry about the code. Good programmers worry about data structures and their relationships.

Linus Tolvards



Benchmarks

16 Java Collections

List
ArrayList
Vector
Collections.syncList()
CopyOnWriteArrayList

Set
LinkedHashSet
Collections.syncSet()
CopyOnWriteArraySet
ConcurrentSkipListSet
ConcurrentHashSet
ConcurrentHashSetV8

Map
LinkedHashMap
Hashtable
Collections.syncMap()
ConcurrentSkipListMap
ConcurrentHashMap
ConcurrentHashMapV8

16 Java Collections

List
ArrayList
Vector
Collections.syncList()
CopyOnWriteArrayList

Set
LinkedHashSet
Collections.syncSet()
CopyOnWriteArraySet
ConcurrentSkipListSet
ConcurrentHashSet
ConcurrentHashSetV8

Map
LinkedHashMap
Hashtable
Collections.syncMap()
ConcurrentSkipListMap
ConcurrentHashMap
ConcurrentHashMapV8

Non thread-safe

Thread-safe

16 Java Collections

List
ArrayList
Vector
Collections.syncList()
CopyOnWriteArrayList

Set
LinkedHashSet
Collections.syncSet()
CopyOnWriteArraySet
ConcurrentSkipListSet
ConcurrentHashSet
ConcurrentHashSetV8

Map
LinkedHashMap
Hashtable
Collections.syncMap()
ConcurrentSkipListMap
ConcurrentHashMap
ConcurrentHashMapV8

x 3 Operations

Traversal	Insertion	Removal

2 Real-world Benchmarks

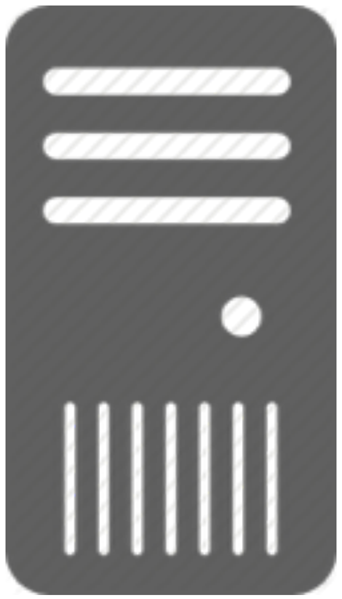
Tomcat

- A web server
- More than 170K lines of Java code
- More than 300 Hashtables

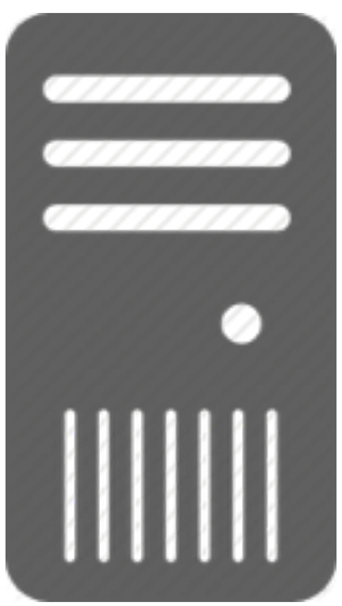
Xalan

- Parses XML in HTML documents
- More than 188K lines of Java code
- More than 140 Hashtables

Experimental Environments

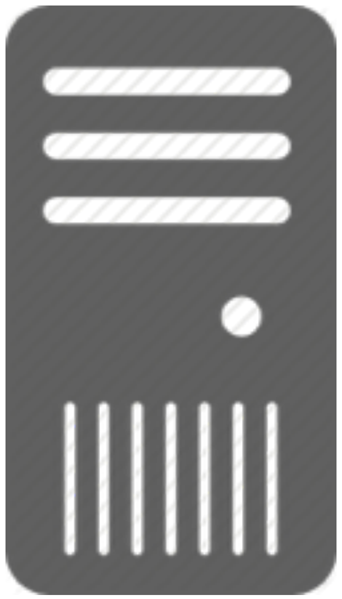


AMD CPU: A 2×16-core, running Debian, 2.4 GHz, 64GB of memory, JDK version 1.7.0 11, build 21.

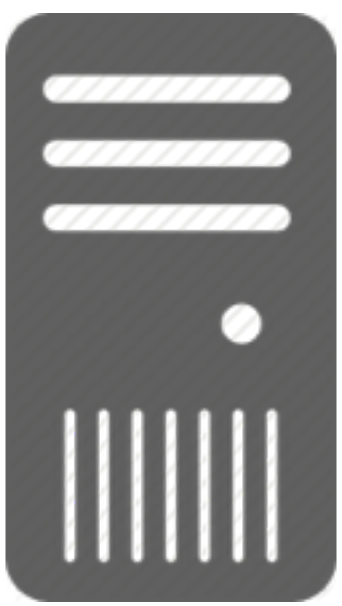


Intel CPU: A 2×8-core (32-cores w/ hyper-threading), running Debian, 2.60GHz, with 64GB of memory, JDK version 1.7.0 71, build 14.

Experimental Environments



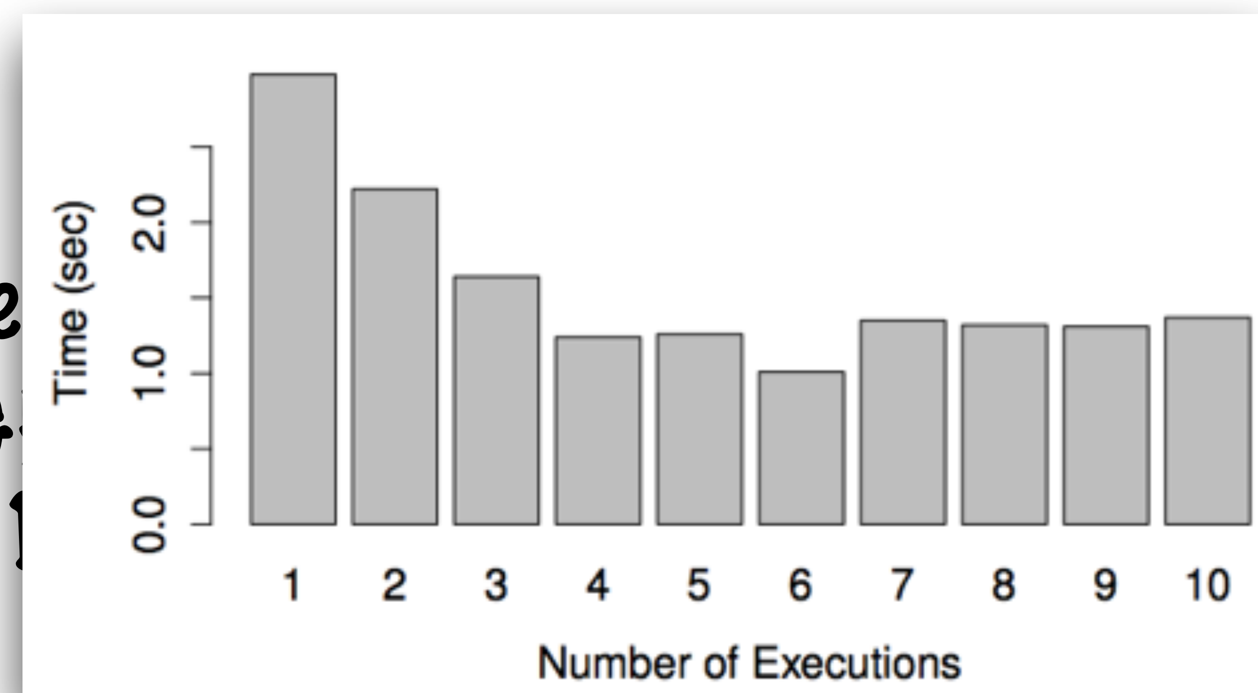
AMD CPU: A 2×16-core, running Debian, 2.4 GHz, 64GB of memory, JDK version 1.7.0 11, build 21.



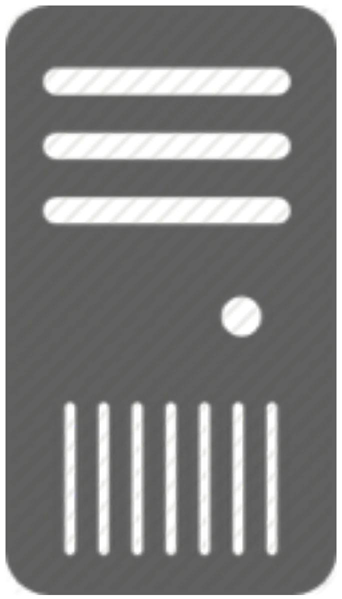
Intel CPU: A 2×8-core

running Debian, 2.60G

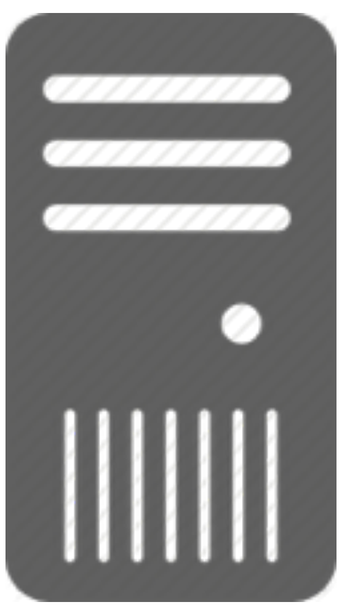
version 1.7.0 71, build 1



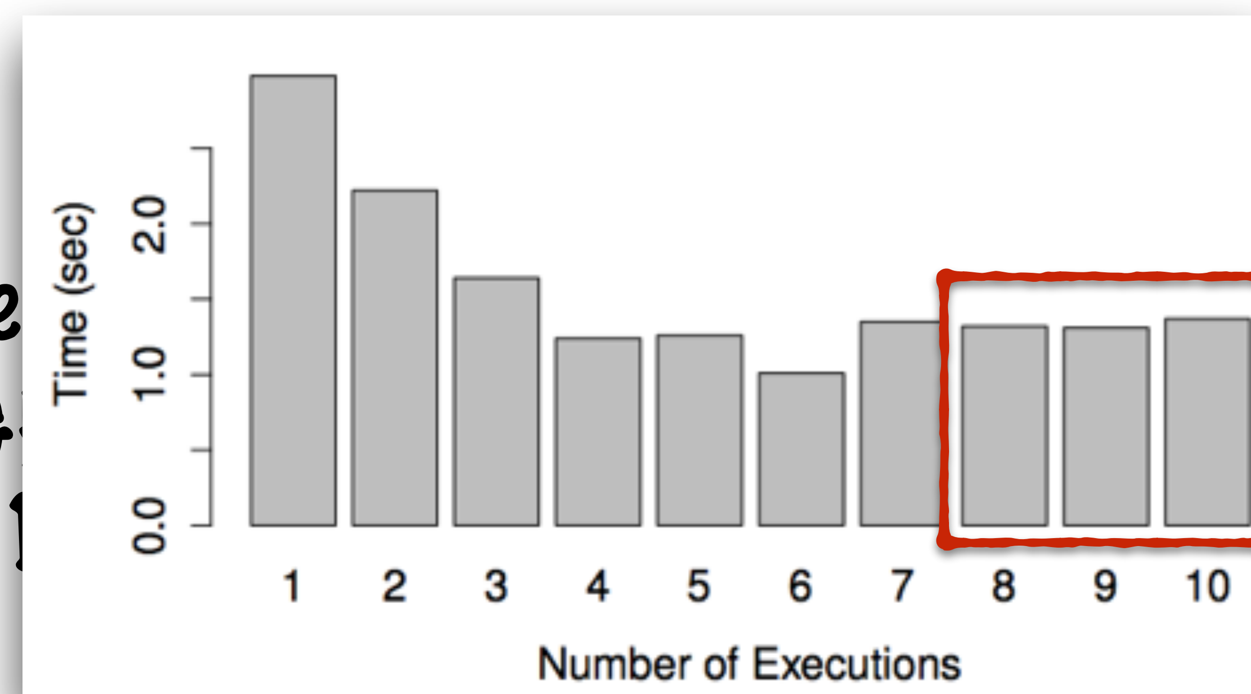
Experimental Environments

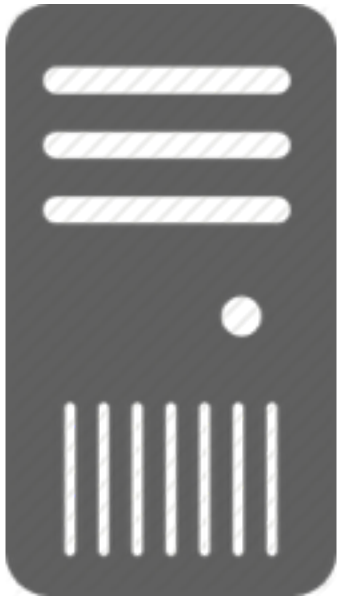


AMD CPU: A 2×16-core, running Debian, 2.4 GHz, 64GB of memory, JDK version 1.7.0 11, build 21.

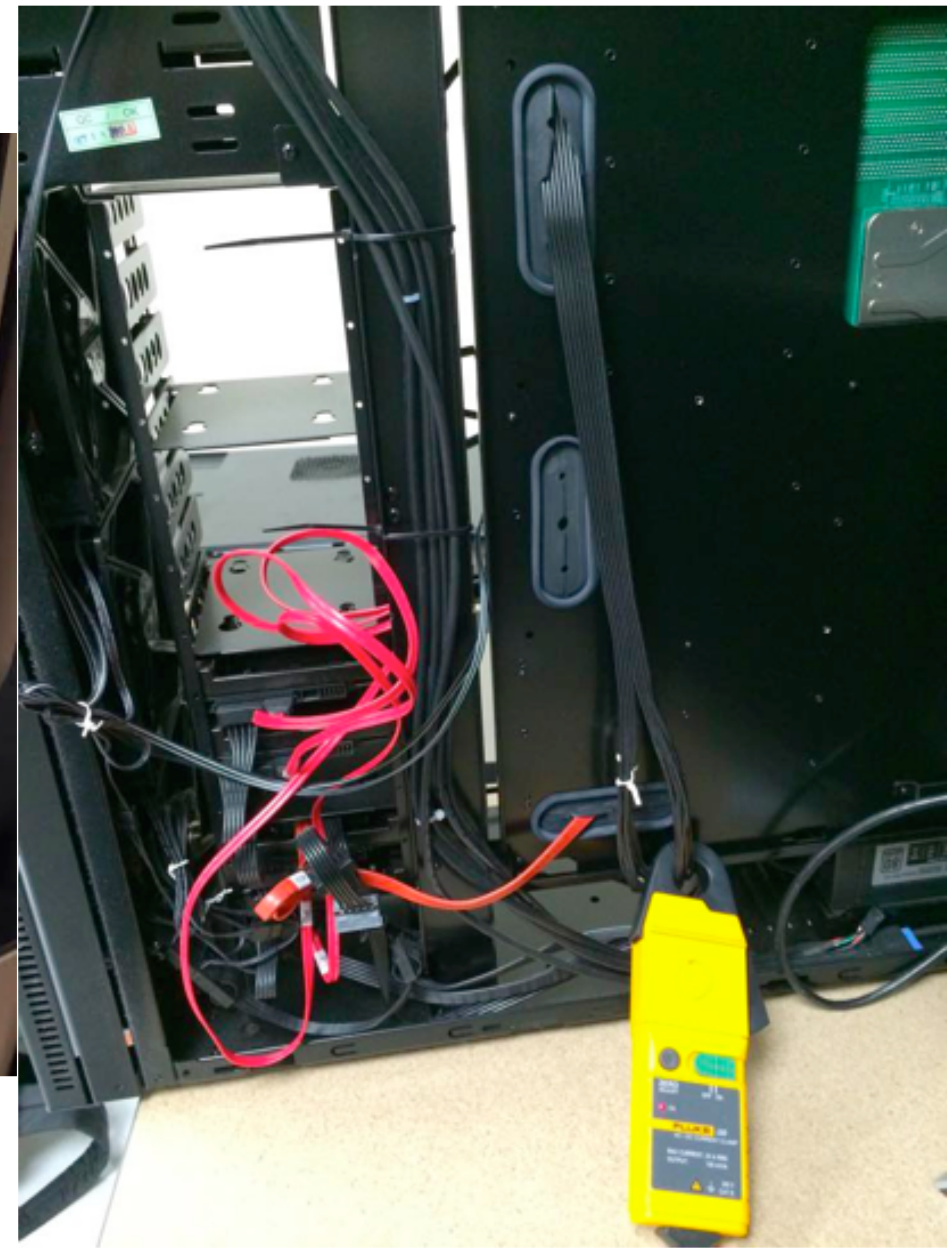


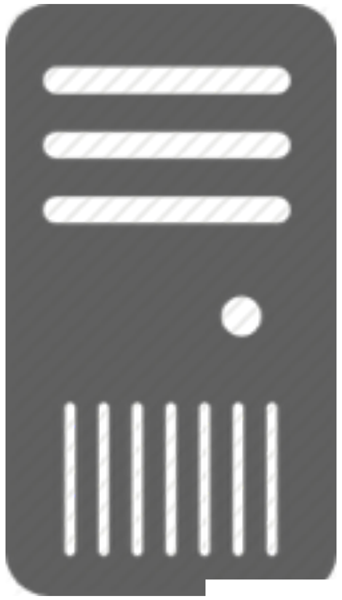
Intel CPU: A 2×8-core
running Debian, 2.60G
version 1.7.0 71, build



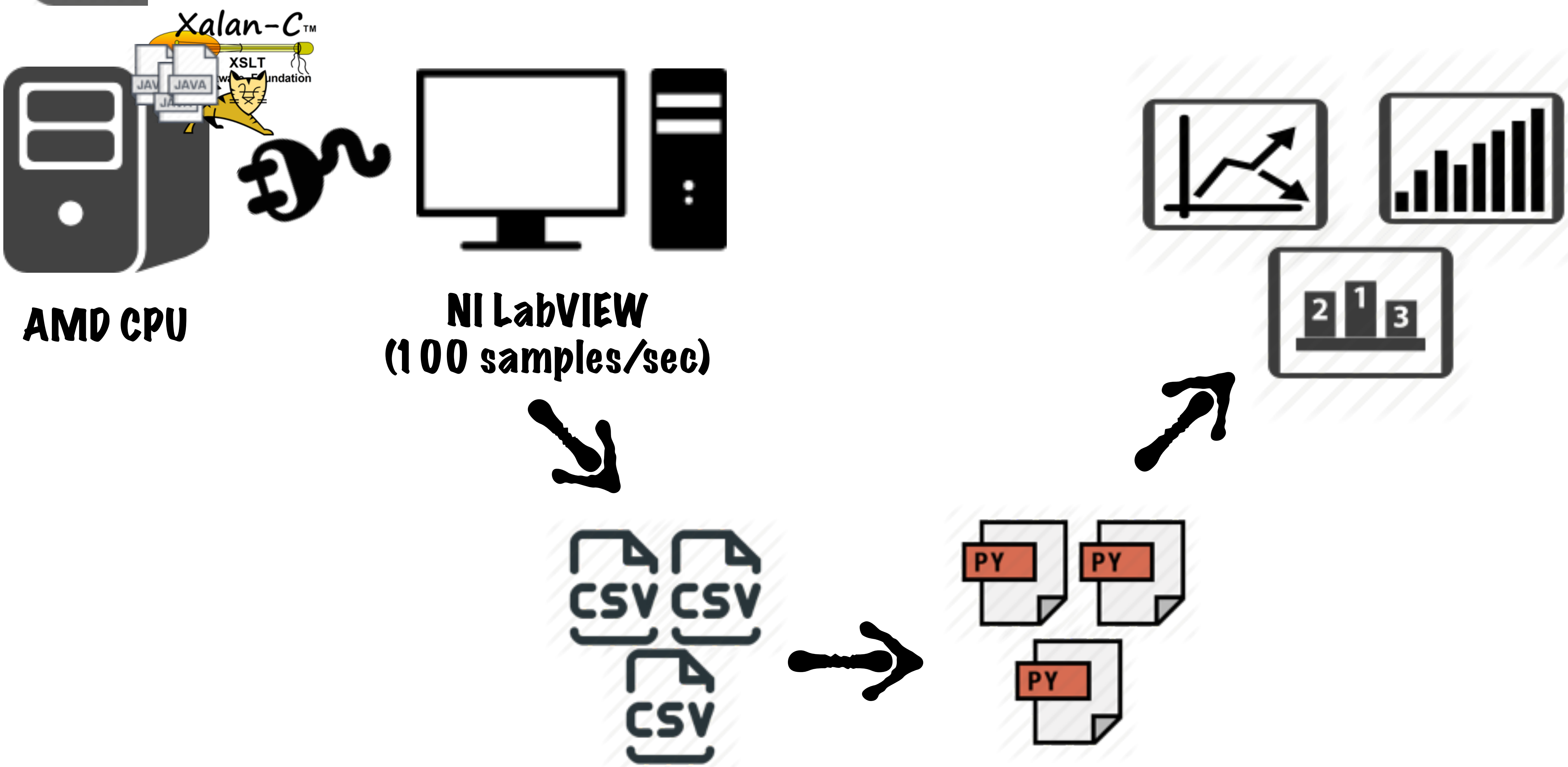


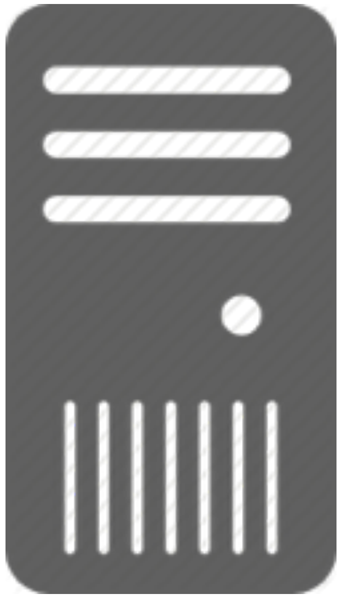
AMD CPU: A 2×16-core, running Debian, 2.4 GHz, 64GB of memory, JDK version 1.7.0 11, build 21.





AMD CPU: A 2×16-core, running Debian, 2.4 GHz, 64GB of memory, JDK version 1.7.0 11, build 21.





Intel CPU: A 2×8-core (32-cores w/ hyper-threading),running Debian, 2.60GHz, with 64GB of memory, JDK version 1.7.0 71, build 14.

jRAPL -- A framework for profiling energy consumption of Java programs

What is jRAPL?

jRAPL is framework for profiling Java programs running on CPUs with Running Average Power Limit (RAPL) support.

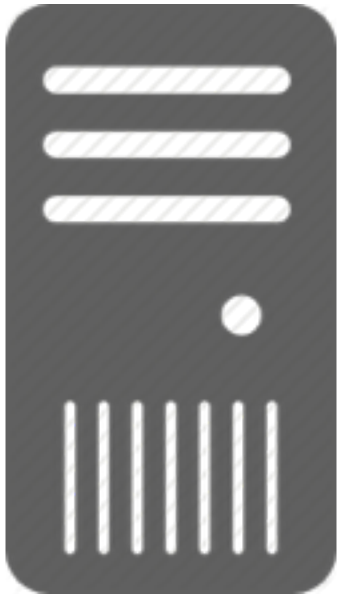
But, what is RAPL?

RAPL is a set of low-level interfaces with the ability to monitor, control, and get notifications of energy and power consumption data of different hardware levels.

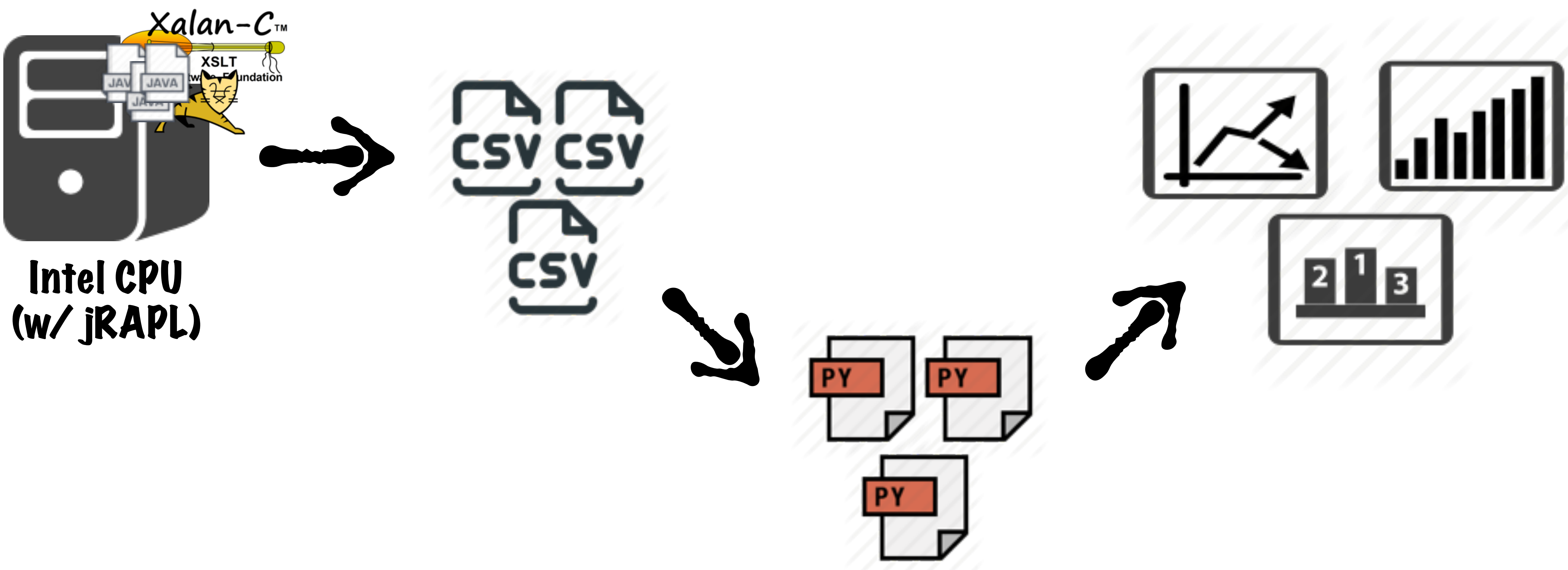
Originally designed by Intel for enabling chip-level power management, RAPL is widely supported in today's Intel architectures, including Xeon server-level CPUs and the popular i5 and i7.

```
double beginning = EnergyCheck.statCheck();
doWork();
double end = EnergyCheck.statCheck();
```

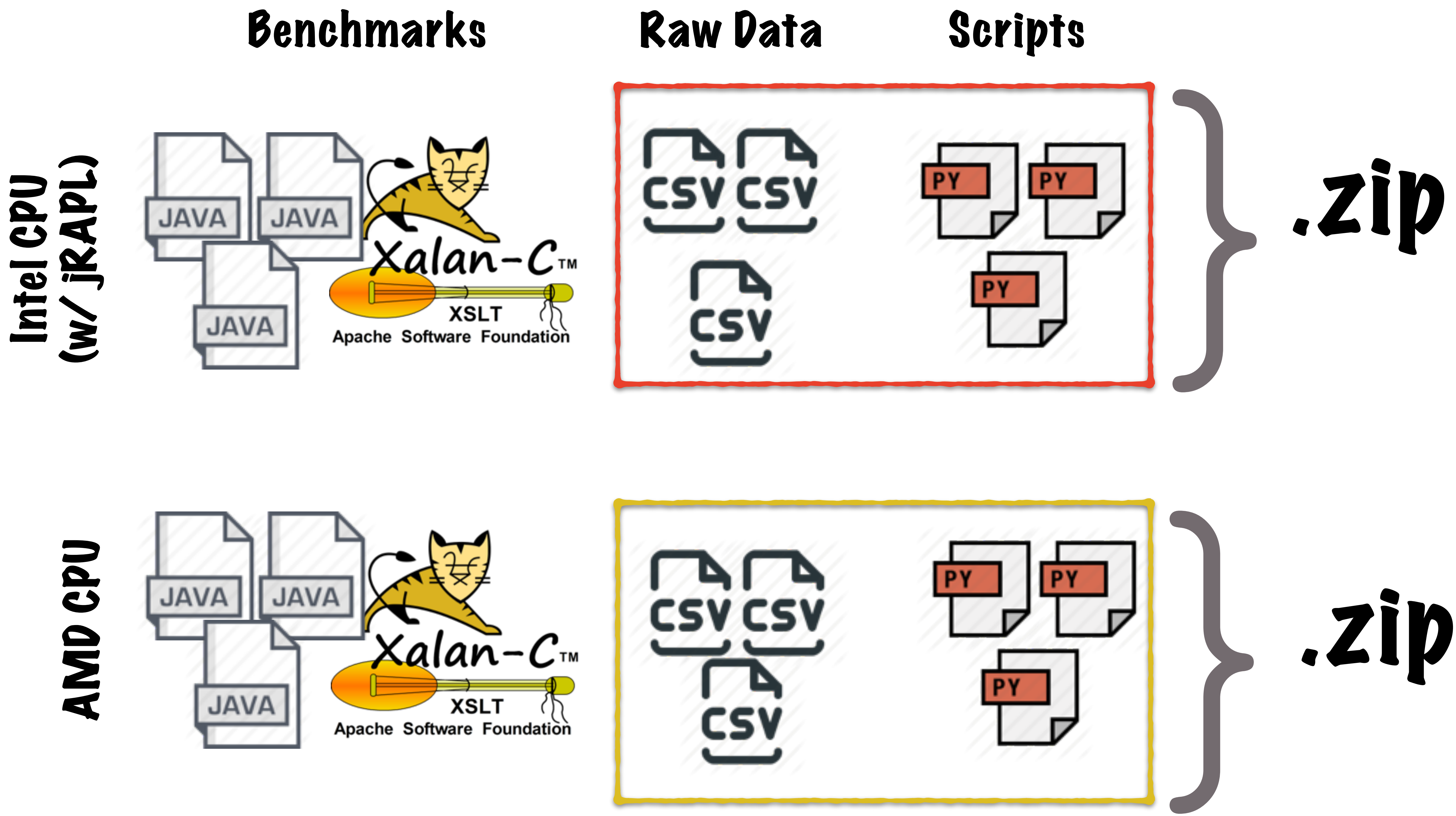
K. Liu, G. Pinto, and D. Liu, "Data-oriented characterization of application-level energy optimization," in Proceedings of the 18th International Conference on Fundamental Approaches to Software Engineering, ser. FASE'15, 2015.



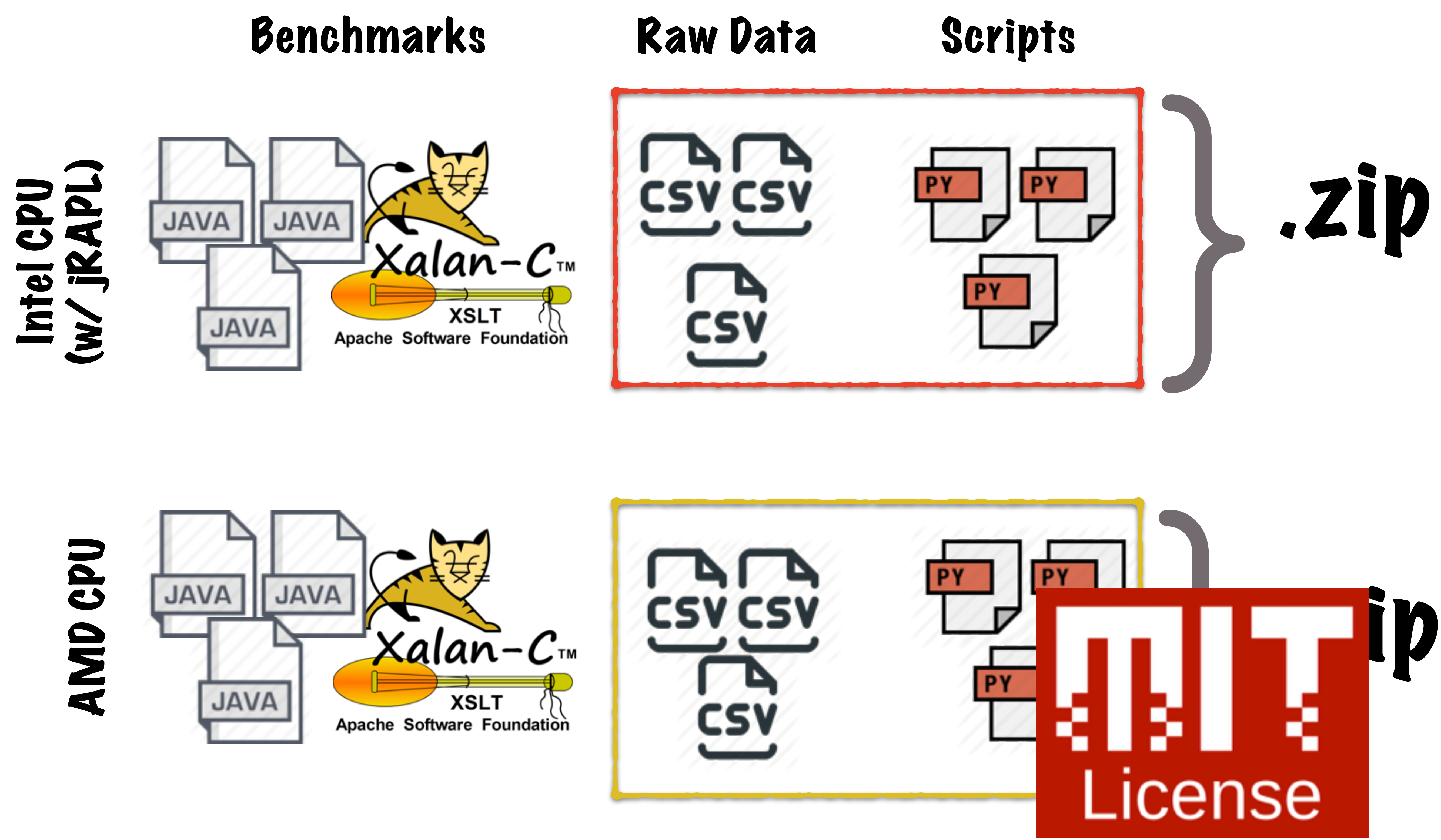
Intel CPU: A 2×8-core (32-cores w/ hyper-threading),running Debian, 2.60GHz, with 64GB of memory, JDK version 1.7.0 71, build 14.



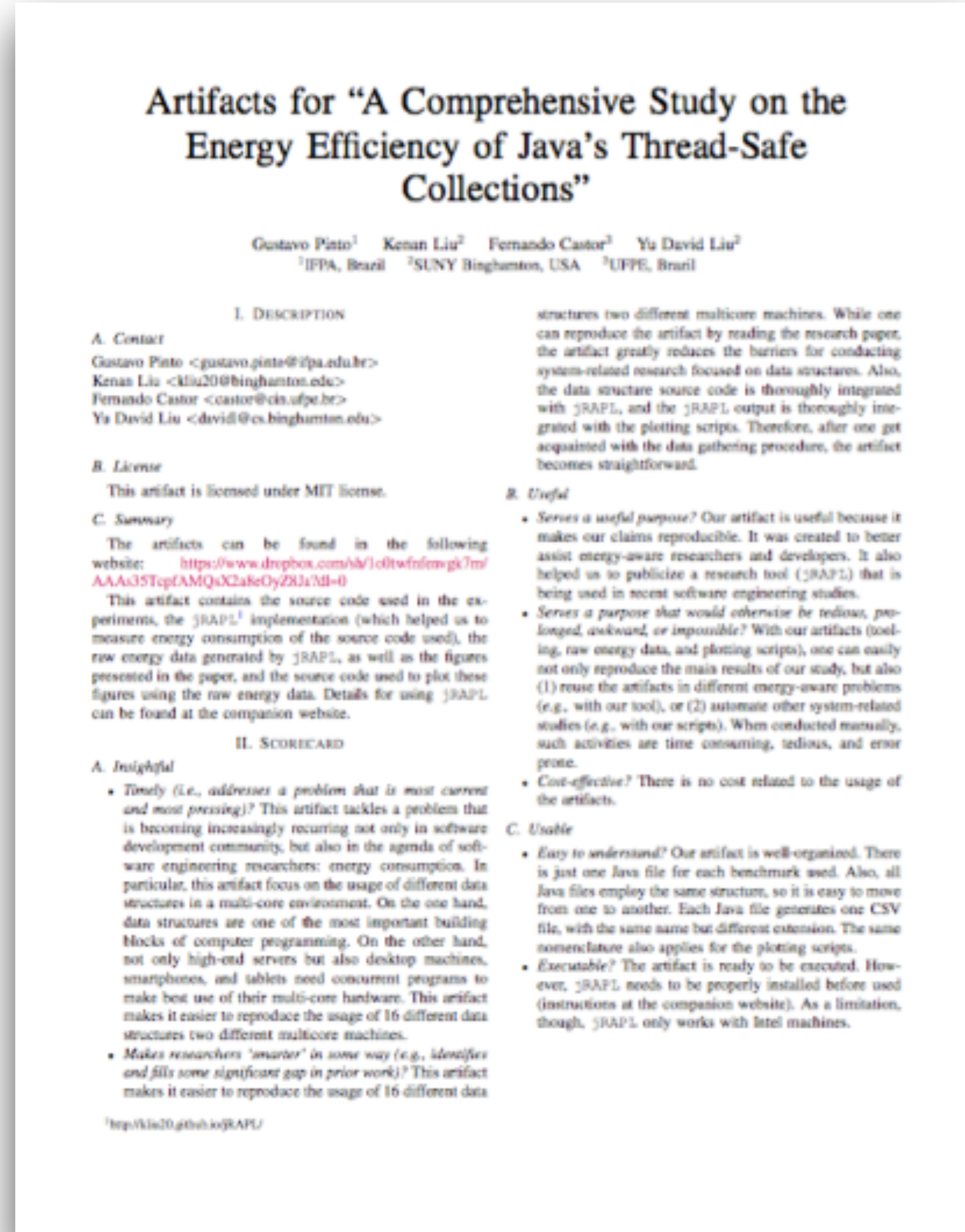
The Artifacts



The Artifacts



More details?


Artifacts for "A Comprehensive Study on the Energy Efficiency of Java's Thread-Safe Collections"

Gustavo Pinto[1] Kenan Liu[2] Fernando Castor[3] Yu David Liu[2]
[1]IFPA, Brazil [2]SUNY Binghamton, USA [3]UFPE, Brazil

I. Description

A. Contact

B. License

This artifact is licensed under MIT license.

C. Summary

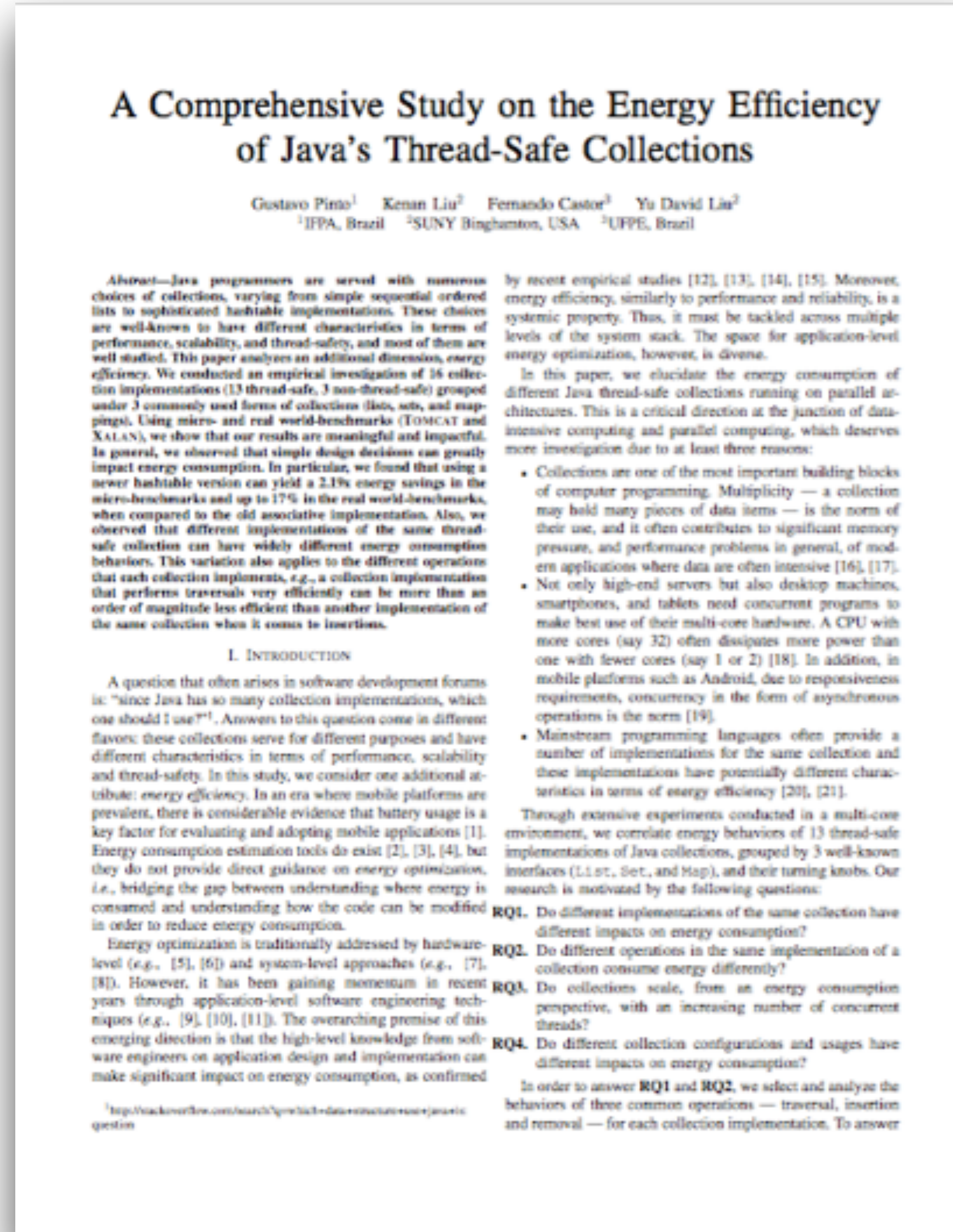
II. Scorecard

A. Insightful

B. Useful

C. Usable

Artifacts Paper


A Comprehensive Study on the Energy Efficiency of Java's Thread-Safe Collections

Gustavo Pinto[1] Kenan Liu[2] Fernando Castor[3] Yu David Liu[2]
[1]IFPA, Brazil [2]SUNY Binghamton, USA [3]UFPE, Brazil

I. Introduction

Research Paper

Characterizing the Energy Efficiency of Java's Thread-Safe Collections in a Multi-Core Environment

Gustavo Pinto

Kenan Liu

Fernando Castor

David Liu

INSTITUTO FEDERAL PARÁ

BINGHAMTON UNIVERSITY STATE UNIVERSITY OF NEW YORK

Centro de Informática UFPE

VIRTUS IMPAVIDA

BINGHAMTON UNIVERSITY STATE UNIVERSITY OF NEW YORK

jRAPL(Java Running Average Power Limit)