Java… to unlock GPU acceleration for Polyglot Language Runtimes

FOSDEM 2024 – Free Java devroom

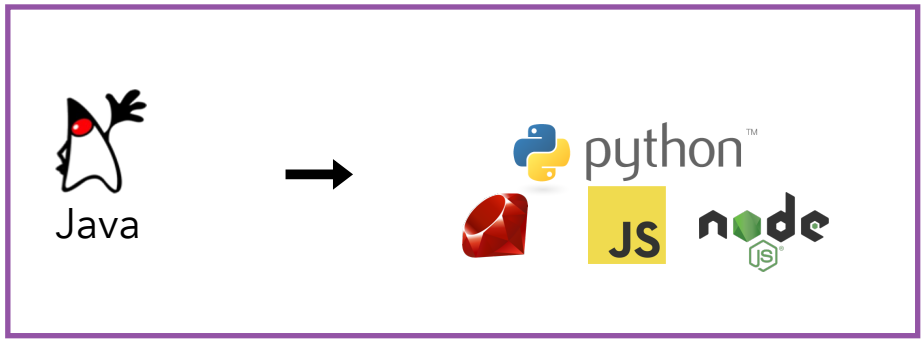
Thanos Stratikopoulos

@thanos_str

https://www.linkedin.com/in/stratika/

3rd February 2024

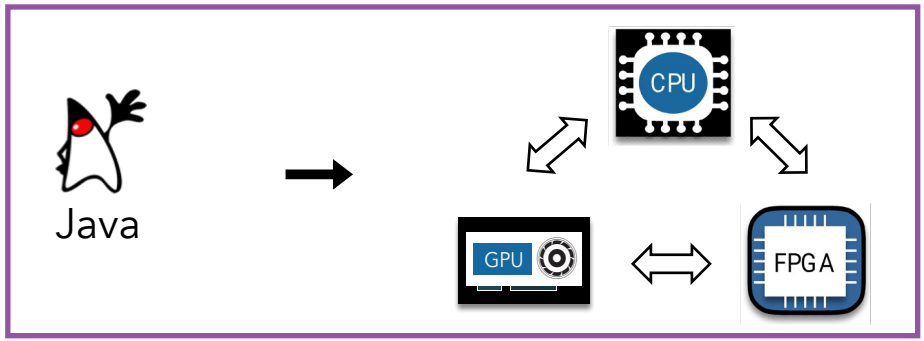
Polyglot Programming?



GraalVM

Truffle Framework

GPU Programming?



TORNADOVM

TornadoVM Resources

SCAN ME

@tornadovm

https://tornadovm.org

https://github.com/beehive-lab/TornadoVM

SCAN ME

A JVM plugin that accelerates Java methods on heterogeneous hardware!

Features:

- Open source
- Platform agnostic
- Automatic code optimization
- Dynamic reconfiguration at runtime
- Off-heap data types (Foreign Function & Memory API)

Using Off-heap Data (since TornadoVM v1.0)

```java
public static void main(String[] args) {

    int size = 512;
    if (args.length >= 1) {
        size = Integer.parseInt(args[0]);
    }

    float[] matrixA = new float[size];

    Random r = new Random();
    IntStream.range(0, size).forEach(idx -> {
        matrixA[idx] = r.nextFloat();
    });
```

```java
public static void main(String[] args) {

    int size = 512;
    if (args.length >= 1) {
        size = Integer.parseInt(args[0]);
    }

    FloatArray matrixA = new FloatArray(size);

    Random r = new Random();
    IntStream.range(0, size).forEach(idx -> {
        matrixA.set(idx, r.nextFloat());
    });
```

https://tornadovm.readthedocs.io/en/latest/offheap-types.html#migrating-from-v0-15-2-to-v1-0

Motivation?

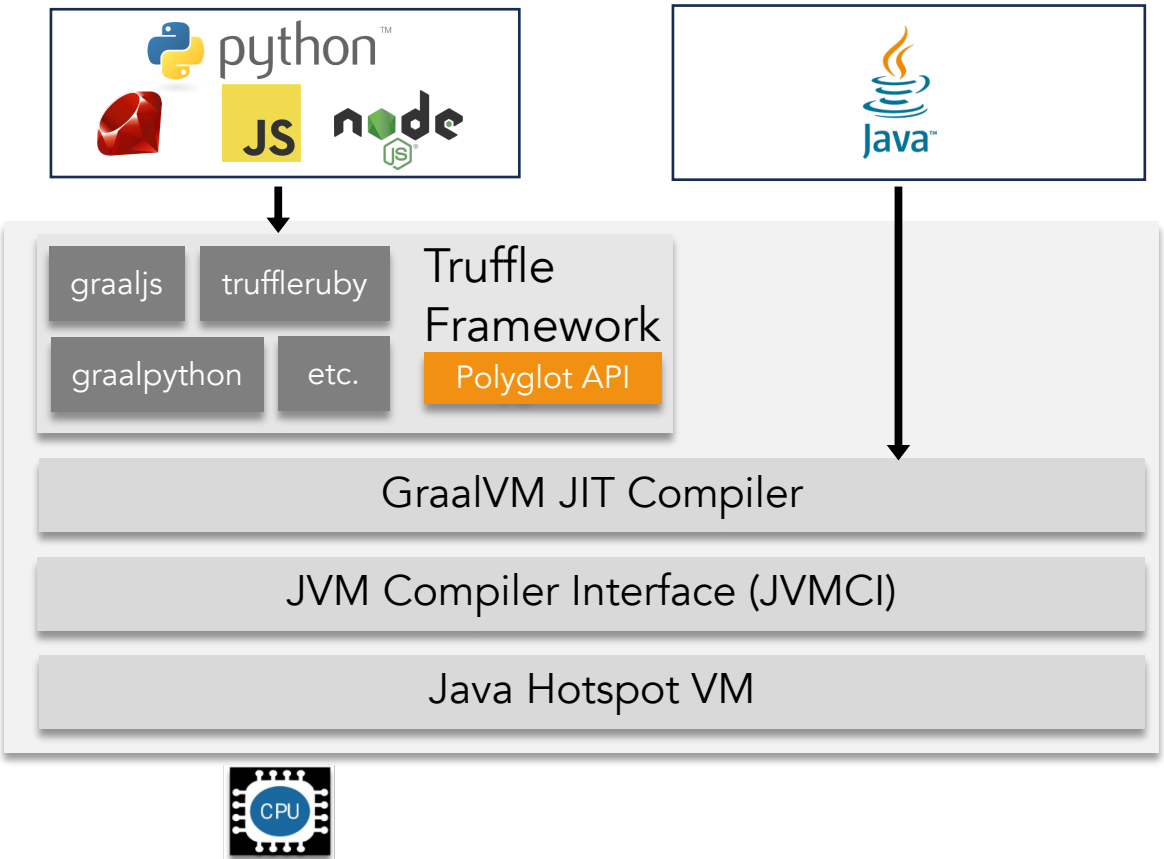
Java *interoperability*
with other PLs via
Polyglot Programming

Hardware acceleration
via a Java API

What if we can create high-performing data science libraries in Java,
to be re-used by other Programming Languages?

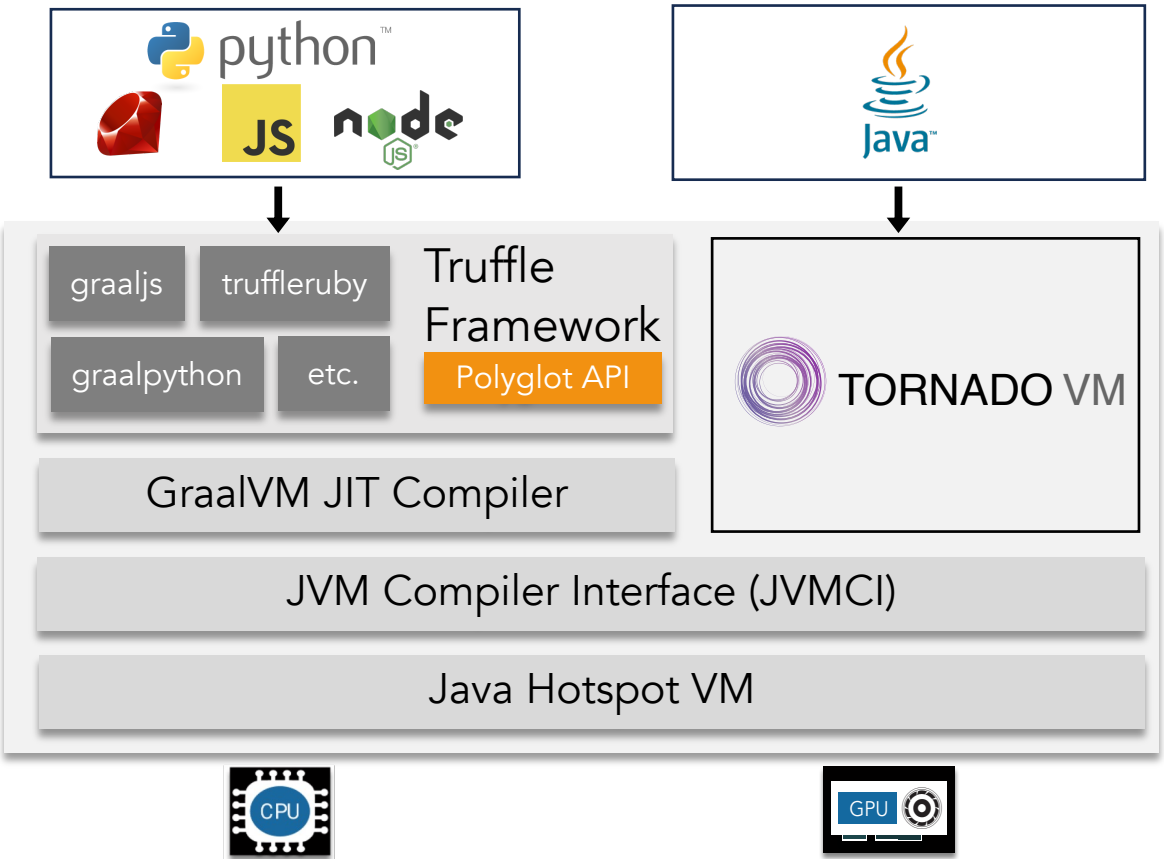
Dive into the
tech flow!

GraalVM Polyglot Runtime Implementations



- ✓ **Java** Interoperability with other PLs!

GraalVM Polyglot Runtimes & TornadoVM

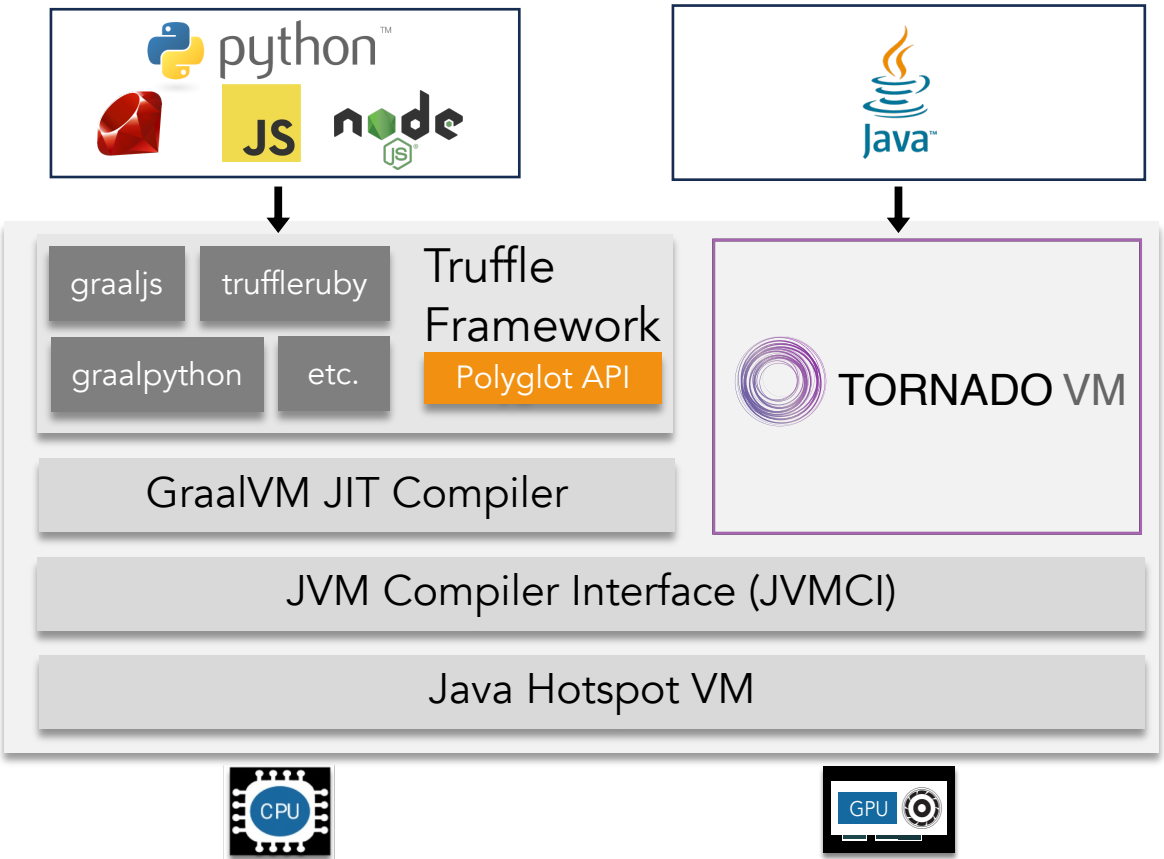


- ✓ **Java** Interoperability with other PLs!
- ✓ **Java** methods specialized for GPUs!

↓

*Dynamic scripting & **accelerated** data science libraries* for **Java** projects.

GraalVM Polyglot Runtimes & TornadoVM



Learn about the APIs!

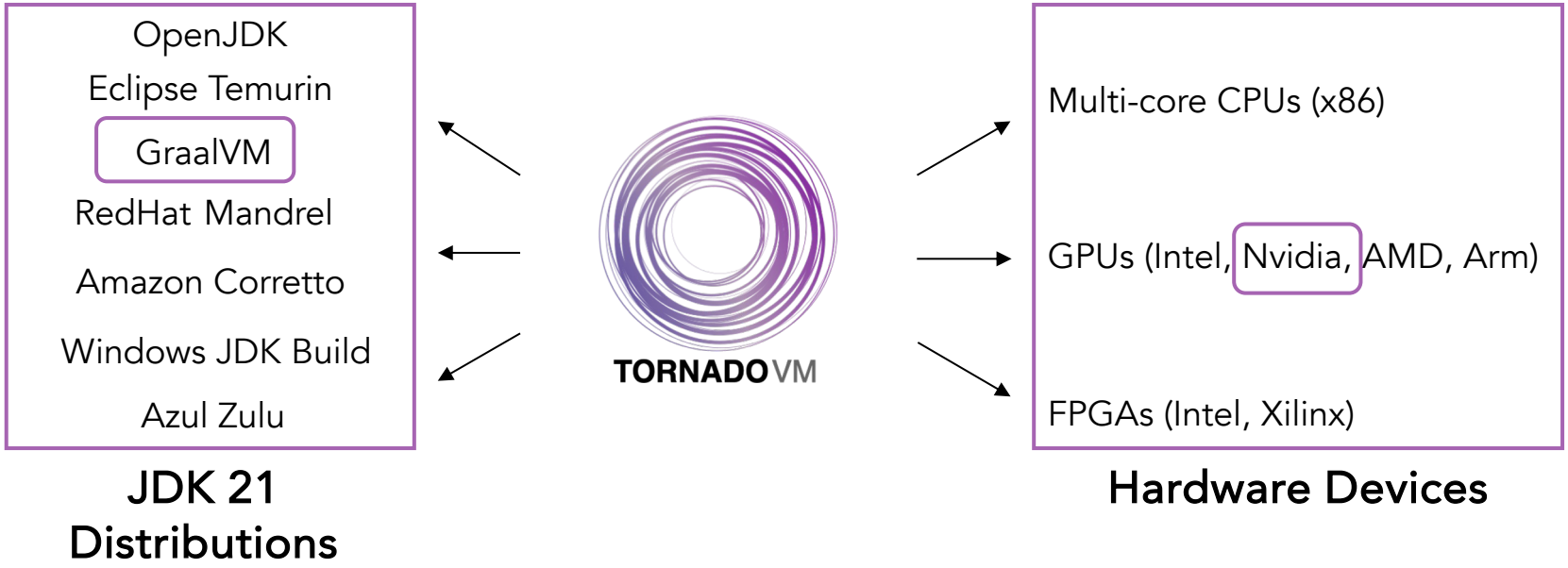
SCAN ME

The exposed Java method can:

- ✓ Use off-heap data types
- ✓ Run with TornadoVM profiler
- ✓ Customize data transfers

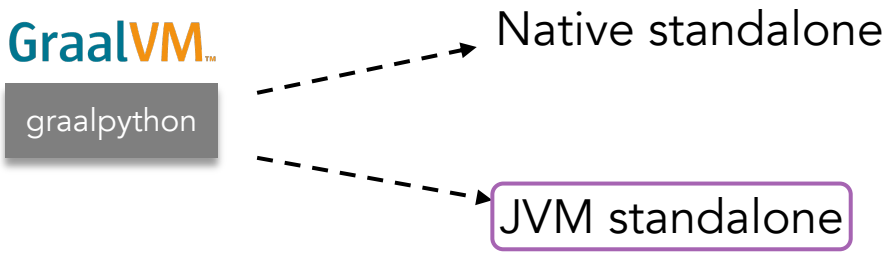
Deployment

TornadoVM Deployment



Plugin to existing JDKs, not a replacement!

GraalVM Deployment



- Oracle GraalVM JDK 21
- Python 3.10 compliant

https://github.com/oracle/graalpython/releases/download/graal-23.1.0/graalpy-community-jvm-23.1.0-linux-amd64.tar.gz

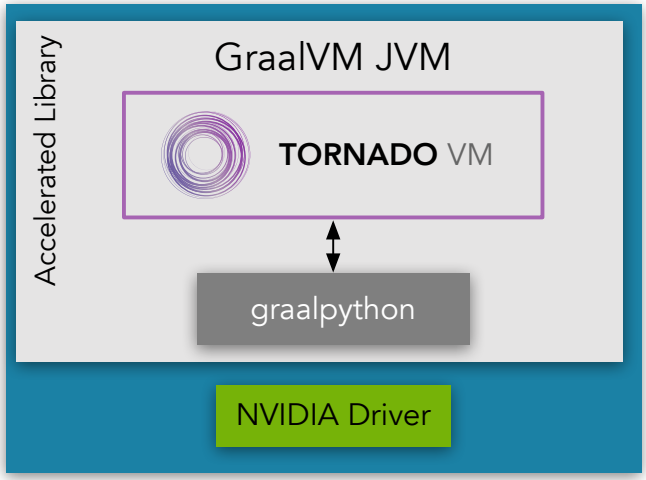
We need the JVM standalone, because **graalpy** will be used by TornadoVM

GraalVM

The JVM shipped with the JVM standalone, is built with *libgraal (modified compiler modules).*

✓ Lower footprint
X Breaks TornadoVM support

Docker Image



Docker image has:

- GraalPy v23.1.0
- TornadoVM 1.0 (my fork)
- NVIDIA GPU Driver

Acceleration Library:
TornadoVM compute examples (Kmeans, etc.)

SCAN ME

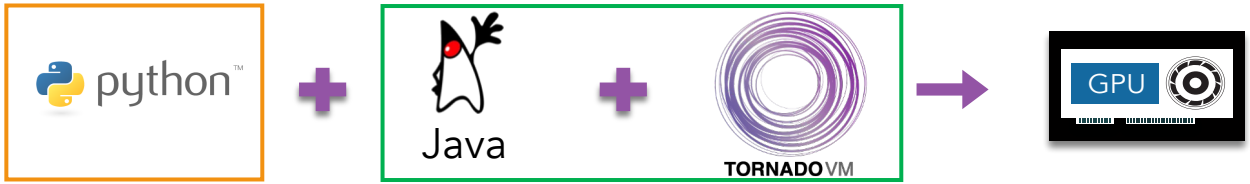
Open-source Dockerfile

SCAN ME

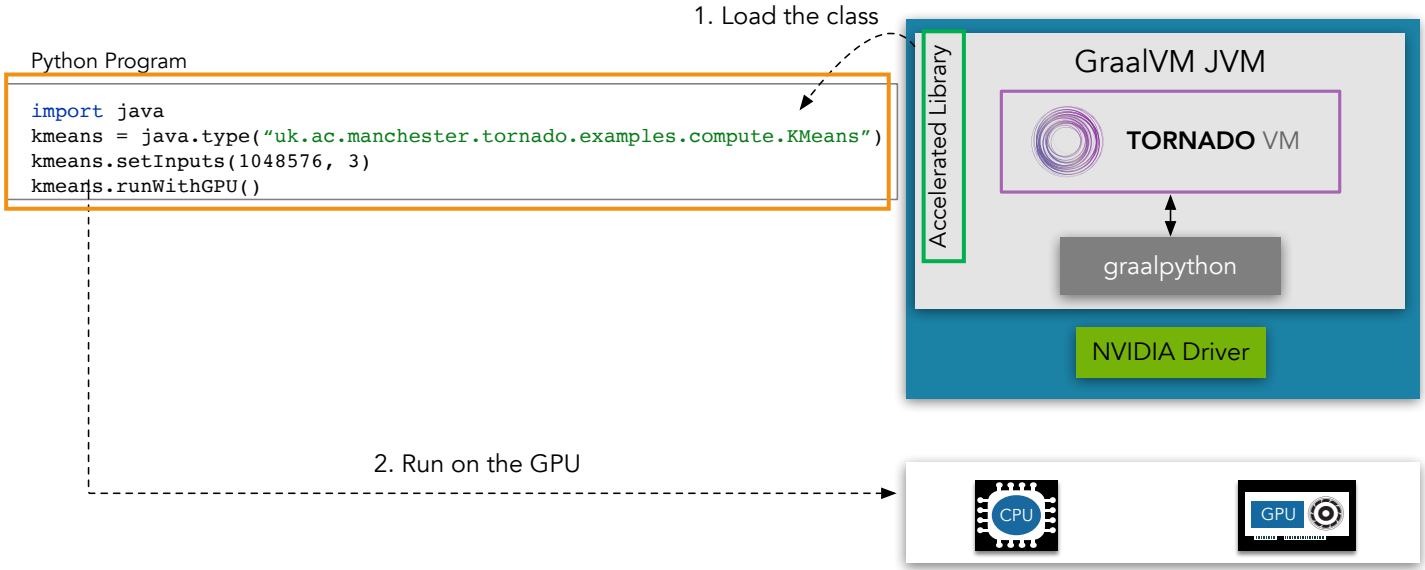
```
$ docker pull beehivelab/tornadovm-polyglot-graalpy-23.1.0-nvidia-opencl-container:fosdem2024
```

Examples

Example



Python invoking Java to compute Matrix Multiplication/Kmeans on GPUs!



Example – Python

```
$ git clone https://github.com/stratika/docker-tornado.git && cd docker-tornado
$ git checkout fosdem2024
$ cat example/polyglot-examples/kmeans.py
#!/usr/bin/python

import java
import time

kmeans = java.type("uk.ac.manchester.tornado.examples.compute.KMeans")

print("Hello World from Python!")

# Set numPoints, numClusters
kmeans.setInputs(1048576, 3)
kmeans.runWithJava()
kmeans.runWithGPU()
```

https://github.com/stratika/TornadoVM/blob/fosdem2024/tornado-examples/src/main/java/uk/ac/manchester/tornado/examples/compute/KMeans.java

Example – Java/TornadoVM

```
public static void setInputs(int numDataPoints, int k) {
    // Cluster the data points
    // Create Data Set: data points
    dataPoints = createDataPoints(numDataPoints);
    centroid = new VectorFloat2(k);

    // Initialize data structures
    clusters = createMatrixOfKClusters(k);
    initializeClusters(k);
}
```

Off-heap data via
Foreign Function & Memory API

Example – Java/TornadoVM

```java
public static Matrix2DInt runWithJava() {
    long start = System.nanoTime();
    assignClusters(dataPoints, clusters, centroid);

    // Recalculate centroids of clusters
    boolean centroidsChanged = true;
    while (centroidsChanged) {
        centroidsChanged = updateCentroids(dataPoints, clusters, centroid);
        if (centroidsChanged) {
            // Reassign data points to clusters
            assignClusters(dataPoints, clusters, centroid);
        }
    }
    long end = System.nanoTime();

    System.out.println("Total time of Java execution: " + (end - start) + " (nanoseconds)");

    return clusters;
}
```

```java
public static Matrix2DInt runWithGPU() {
    // 1. Create a TaskGraph for the assign cluster method
    TaskGraph taskGraph = new TaskGraph( name: "clustering/examples") //
            .transferToDevice(DataTransferMode.FIRST_EXECUTION, clusters, dataPoints) //
            .transferToDevice(DataTransferMode.EVERY_EXECUTION, centroid) //
            .task( id: "kmeans", KMeans::assignClusters, dataPoints, clusters, centroid) //
            .transferToHost(DataTransferMode.EVERY_EXECUTION, clusters);

    // 2. Create an execution plan
    TornadoExecutionPlan executionPlan = new TornadoExecutionPlan(taskGraph.snapshot());

    // 3. Execute the plan - KMeans Clustering Algorithm
    long start = System.nanoTime();
    executionPlan.execute();

    // 3.1 Recalculate centroids of clusters while the centroids list change between iterations.
    boolean centroidsChanged = true;
    while (centroidsChanged) {
        // Recalculate centroids. The following method is executed on a CPU (without TornadoVM).
        centroidsChanged = updateCentroids(dataPoints, clusters, centroid);
        if (centroidsChanged) {
            // If there are new changes, then the clusters are re-assigned
            executionPlan.execute();
        }
    }
    long end = System.nanoTime();
    System.out.println("Total time of TornadoVM execution: " + (end - start) + " (nanoseconds)");

    return clusters;
}
```

Example - Running

```
$ docker pull beehivelab/tornadovm-polyglot-graalpy-23.1.0-nvidia-opencl-container:fosdem2024
$
$ ./polyglotImages/polyglot-graalpy/tornadovm-polyglot.sh tornado --truffle python
example/polyglot-examples/kmeans.py
$
$ ./polyglotImages/polyglot-graalpy/tornadovm-polyglot.sh tornado --truffle python
example/polyglot-examples/mxmWithTornadoVM.py
```


thanos@thanos-Precision-5570: ~
thanos@thanos-Precision-5570:~$

Not only Python, but also JS, Ruby


```
$ docker pull beehivelab/tornadovm-polyglot-graalpy-23.1.0-nvidia-opencl-container:fosdem2024
$ ./polyglotImages/polyglot-graalpy/tornadovm-polyglot.sh tornado --printKernel --truffle python
example/polyglot-examples/mxmWithTornadoVM.py
```


```
$ docker pull beehivelab/tornadovm-polyglot-graaljs-23.1.0-nvidia-opencl-container:latest
$ ./polyglotImages/polyglot-graaljs/tornadovm-polyglot.sh tornado --printKernel --truffle js
example/polyglot-examples/mxmWithTornadoVM.js
```

```
$ docker pull beehivelab/tornadovm-polyglot-truffleruby-23.1.0-nvidia-opencl-container::latest
$ ./polyglotImages/polyglot-truffleruby/tornadovm-polyglot.sh tornado --printKernel --truffle ruby
example/polyglot-examples/mxmWithTornadoVM.rb
```

https://www.tornadovm.org/post/hardware-acceleration-for-polyglot-runtimes
https://tornadovm.readthedocs.io/en/latest/truffle-languages.html

Summary

Key Takeaways

- GraalVM & Truffle enable Java to interoperate with other PLs
- TornadoVM offloads Java methods for hardware acceleration
- TornadoVM offers a Java API & new off-heap types.

It is possible to create high-performing data science libraries in Java, and re-use them by other Programming Languages!

Our Team

<u>Academic Staff</u>

- Christos Kotselidis

<u>Research Staff</u>

- Juan Fumero
- Thanos Stratikopoulos
- Maria Xekalaki
- Michail Papadimitriou
- Orion Papadakis

<u>Alumni</u>

- James Clarkson
- Foivos Zakkak
- Benjamin Bell
- Amad Aslam
- Ales Kubicek
- Florin Blanaru
- Gyorgy Rethy
- Mihai-Christian Olteanu
- Ian Vaughan
- Tianyu Zuo

Join our community

TORNADOVM

@tornadovm

http://tornadovmcommunity.slack.com/

https://github.com/beehive-lab/TornadoVM

This work is partially supported by research grants from the
EU Horizon 2020 and EU Horizon Europe research and innovation programme,
UKRI, and Intel Corporation.

ELEGANT

Thank you
Q&A

The University of Manchester

athanasios.stratikopoulos@manchester.ac.uk

@thanos_str

https://www.linkedin.com/in/stratika/