

Mani Sarkar
github: [neomatrix369](https://github.com/neomatrix369)
twitter: [@theNeomatrix369](https://twitter.com/@theNeomatrix369)
blogs: <https://medium.com/@neomatrix369>



Naturally, getting productive

my journey with Grakn and Graql

#grakn**cosmos**

6th Feb 2020



GRAKN COSMOS

About me



Mani Sarkar

**Freelance Software,
Data, ML Engineer**

Java / JVM

**Cloud / Infra /
DevOps**

Polyglot developer

AI / ML / DL / DS / NLP

**LJC, Devoxx,
developer communities**

**JCP member, F/OSS projects:
@adoptopenjdk @graalvm @truffleruby**

**Java Champion, Oracle Groundbreaker Ambassador,
Software Crafter, Blogger, Speaker**

More about me

Introduction

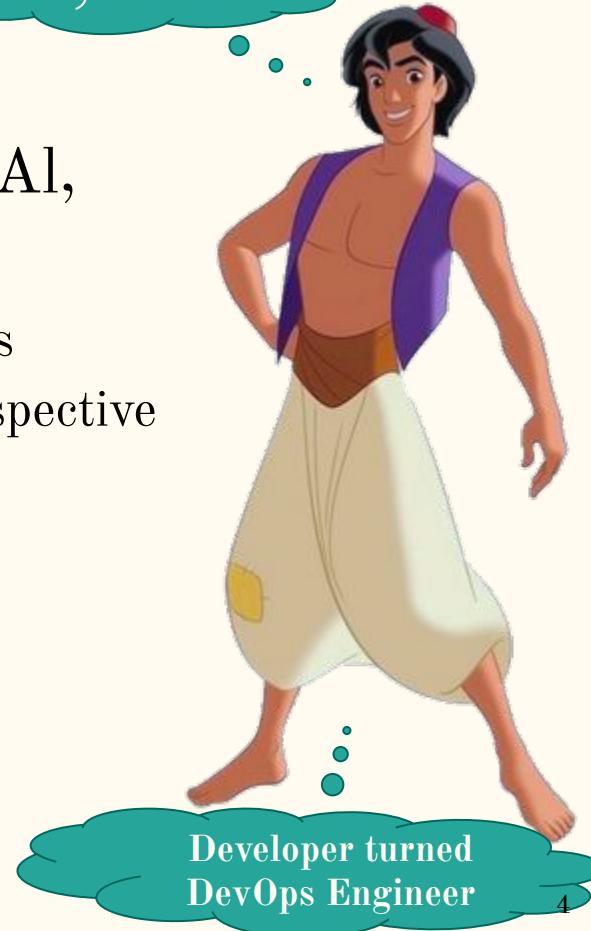
Al, here!

About the talk

- **Productivity with Grakn:** Journey with Al,
Developer turned DevOps Engineer
 - Quick introduction to GraalVM and other JVMs
 - Compare measurements from Performance perspective
 - Benefits
 - Resources on how to apply the above



@theNeomatrix369



About the talk

- **Speaking Graql:** journey with Jas, a Data Scientist, NLP expert
 - **Experimental:** Translate from English to **Graql**
 - **Experimental:** Translate from **Graql** to English
- **Closing:** Summary, Lots of Resources and Q&A



@theNeomatrix369

Presentation slides: live

<https://bit.ly/grakn-graql-graalvm-slides>



Github: <https://bit.ly/grakn-graql-graalvm>

Thank You!

And everyone
else at
GraknLabs

- Grakn Cosmos 2020, it's organisers (Daniel Crowe)
- Good folks at GraknLabs (Haikal, Tomas, Joshua,..)



@theNeomatrix369



A Well Deserved

Applause To

GraknLabs

Thank You!

- Developers from the various developer communities I have been at over the years
- You for being here, sparing your time, and entrusting it with me for this session



@theNeomatrix369



Disclaimer

- ***YMMV***
- My **first GraknLabs** talk at a conference
- Might have rough edges and **inaccuracies**
- Sharing our **learnings** over the past year
- Gathered ideas from **different sources**
- **Sharing ideas and guidelines**, not a silver-bullet
- If it's not clear, **tell me!**
- I'm just a community member, no-strings-attached

Citation

The respective authors and creators are, and remain the true owners of the images and other artifacts used in this presentation.

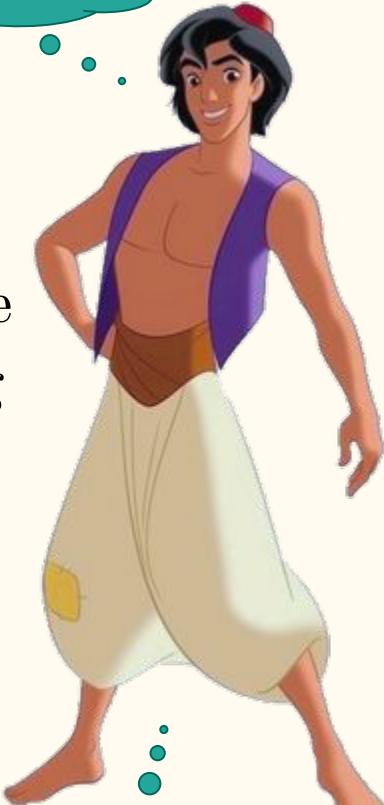
Thank you for your creations!

Conversation with Al

Al, again!

Productivity: Grakn (with Al)

- Al is setting up all things Cloud, infra, DevOps!
- He is keen on productivity, efficiency, and performance
- Also learning pragmatism over trying to do everything
- His high-level goals with Grakn are
 - Quick startup
 - High throughput
 - Low latency
 - High availability
 - Low maintenance
 - Easy knowledge transfer (to learn and to share)



Developer turned
DevOps Engineer

Running Grakn

on any JVM

JVM / JDK 101

Java 8

JDK
from any
Java
vendor

Grakn, Graql, benchmark

Java Virtual Machine (JVM)

Operating System (Windows,
Linux, MacOS)

Hardware (Your Laptop,
Desktop, Server)

I know that
thing! Yeah!

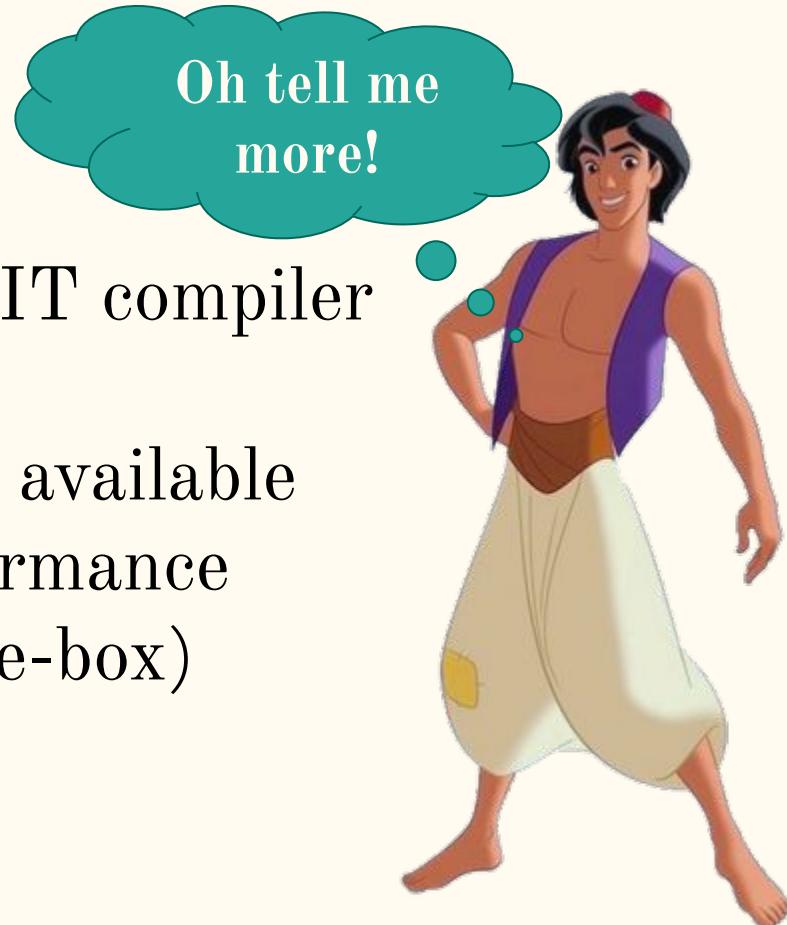


Running Grakn

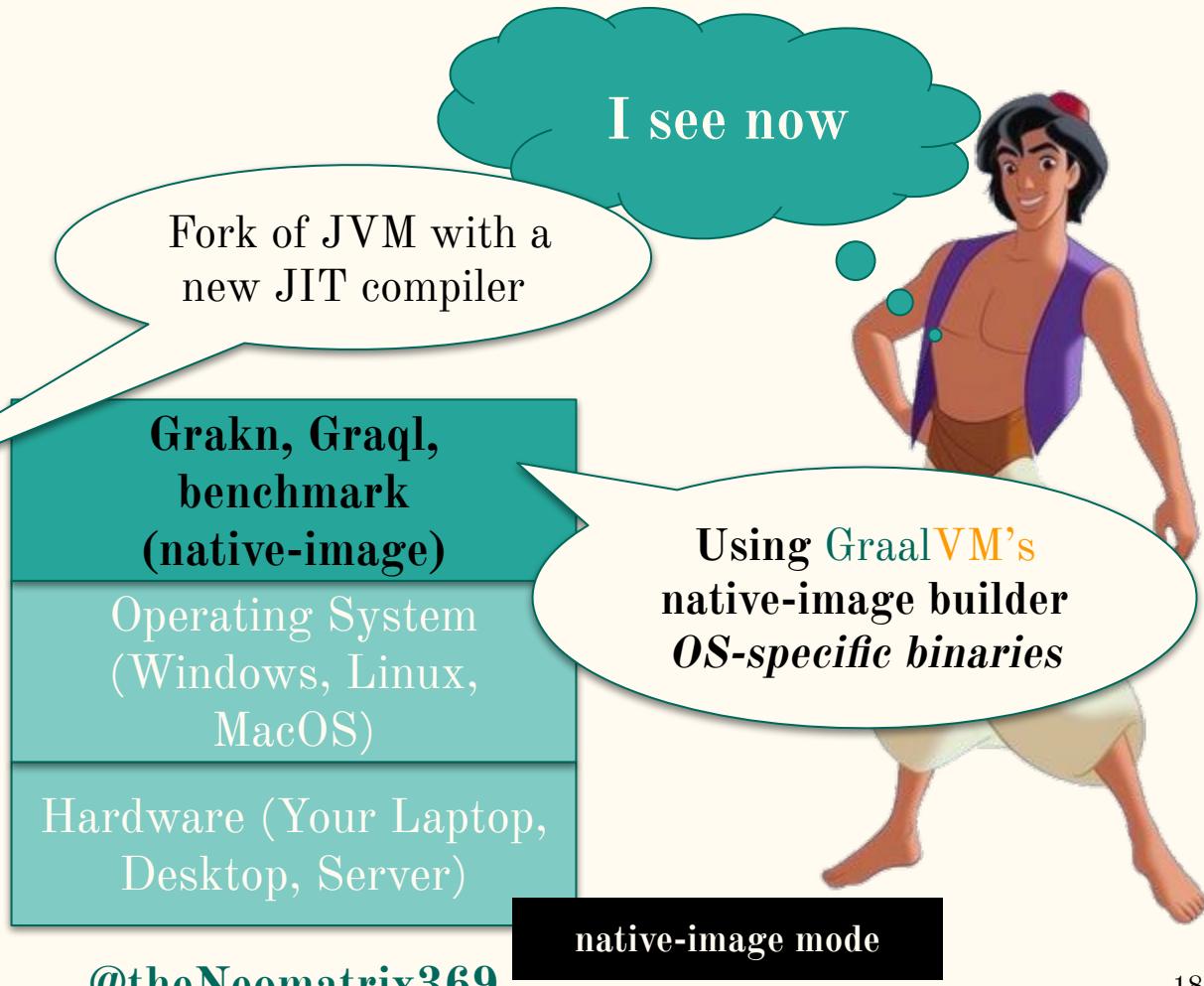
GraalVM

GraalVM 101

- Fork of a JVM with a new JIT compiler
- Built on top of **OpenJDK**
- Java 8 and Java 11 versions available
- Primary purpose: high performance
- JVM mode (works-out-of-the-box)
- native-image mode
- Polyglot VM mode



GraalVM 101

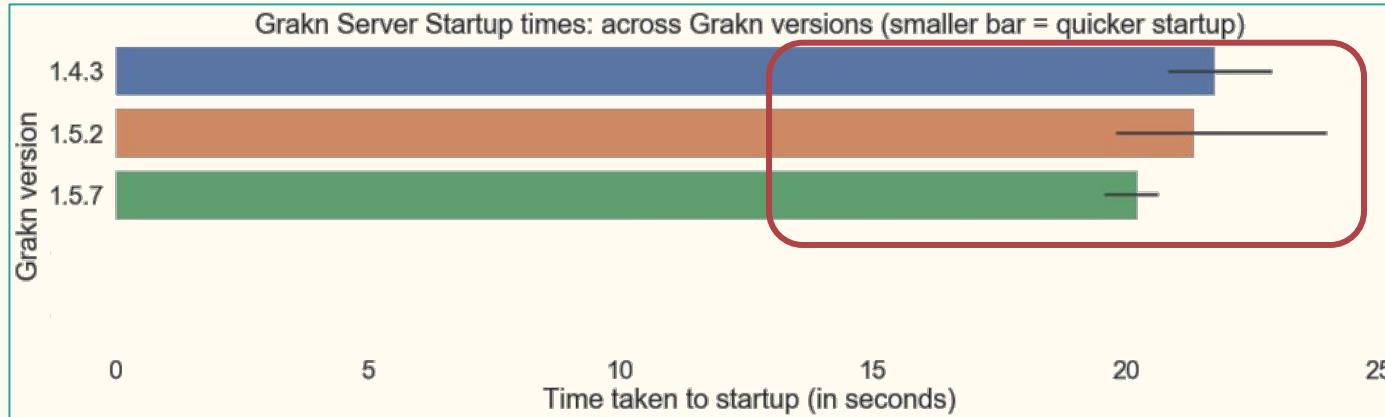


Measuring Grakn startup times

~ early to mid-2019

Grakn: Server startup times

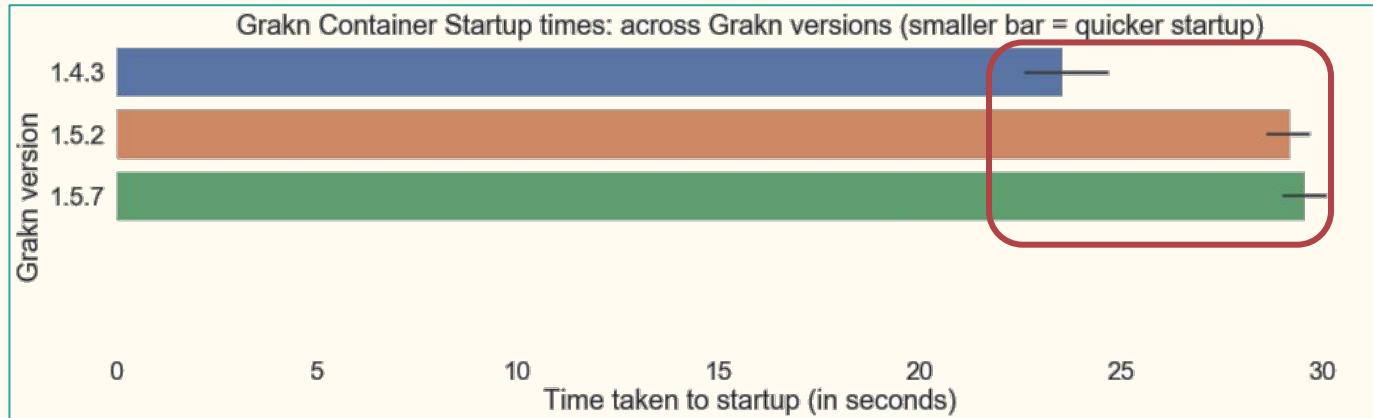
Community
thanks you for
sharing them!



```
$ ./measureTradVersusGraalVMStartupTime.sh
```

Grakn: Container startup times

Hmm! Is that
Grakn server
related?

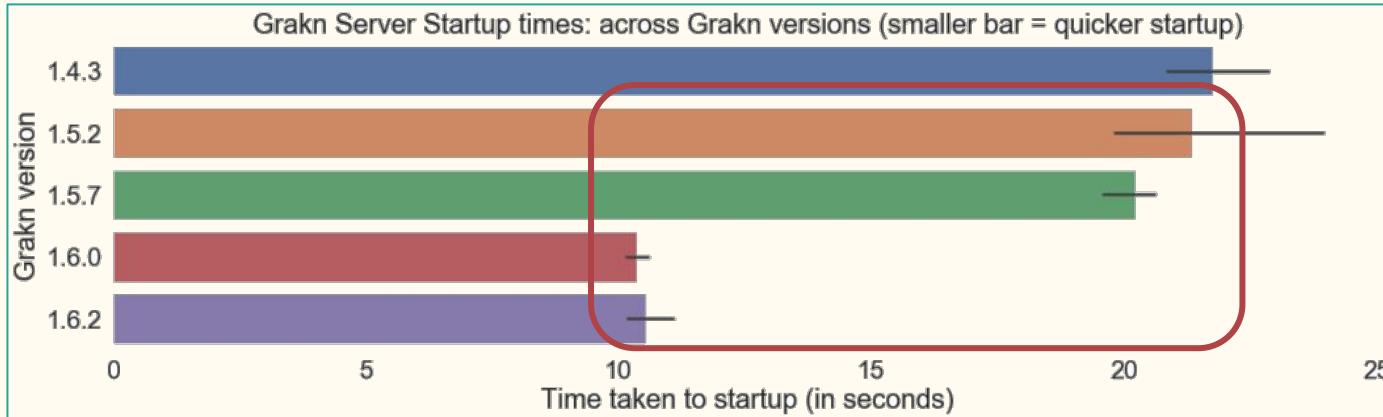


```
$ ./measureTradVersusGraalVMStartupTime.sh
```



Revisited in Feb 2020

Grakn: Server startup times



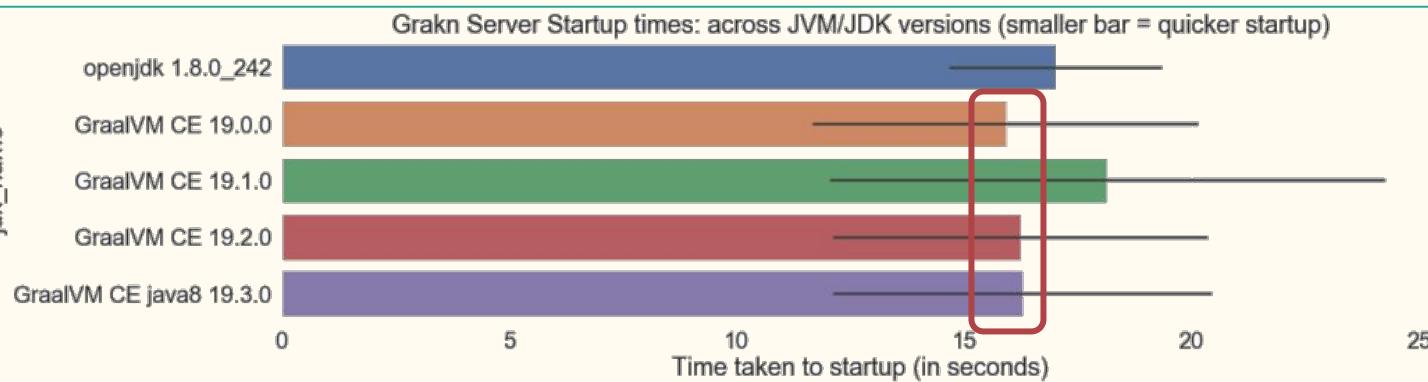
```
$ ./measureTradVersusGraalVMStartupTime.sh
```

Server startup
improved so
much!



Grakn: Server startup times

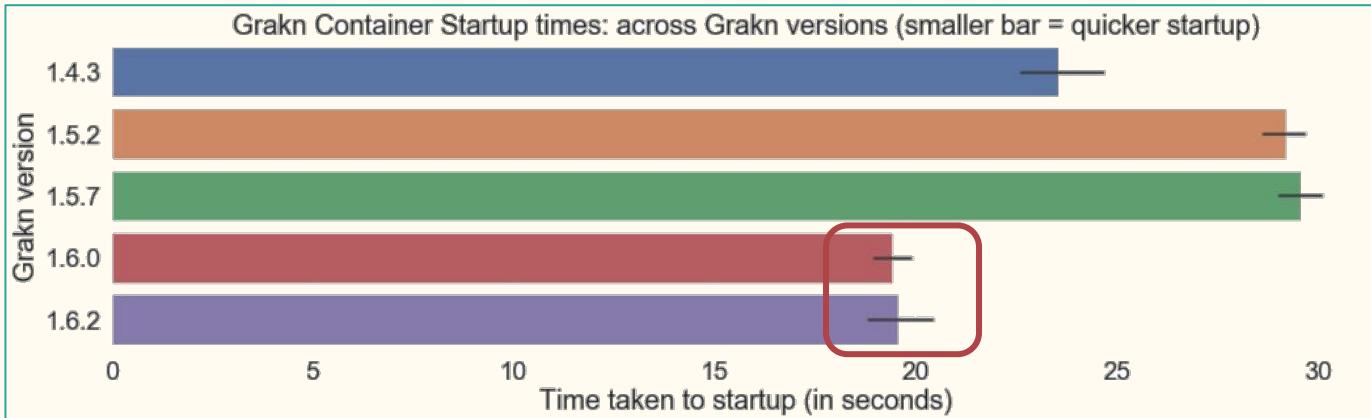
Looking at the trend!



```
$ ./measureTradVersusGraalVMStartupTime.sh
```

Grakn: Container startup times

Container startup also improved!

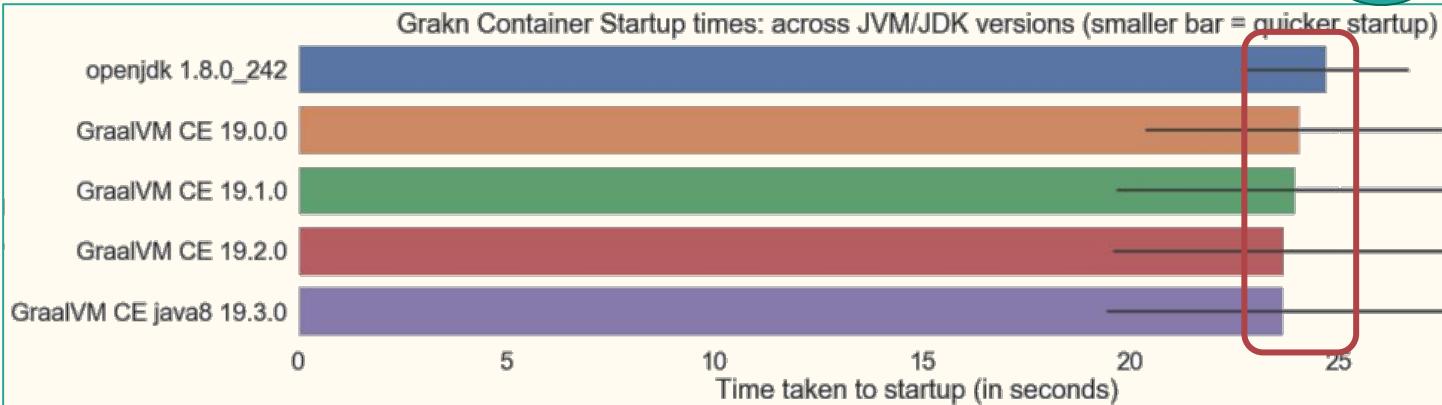


```
$ ./measureTradVersusGraalVMStartupTime.sh
```



Grakn: Container startup times

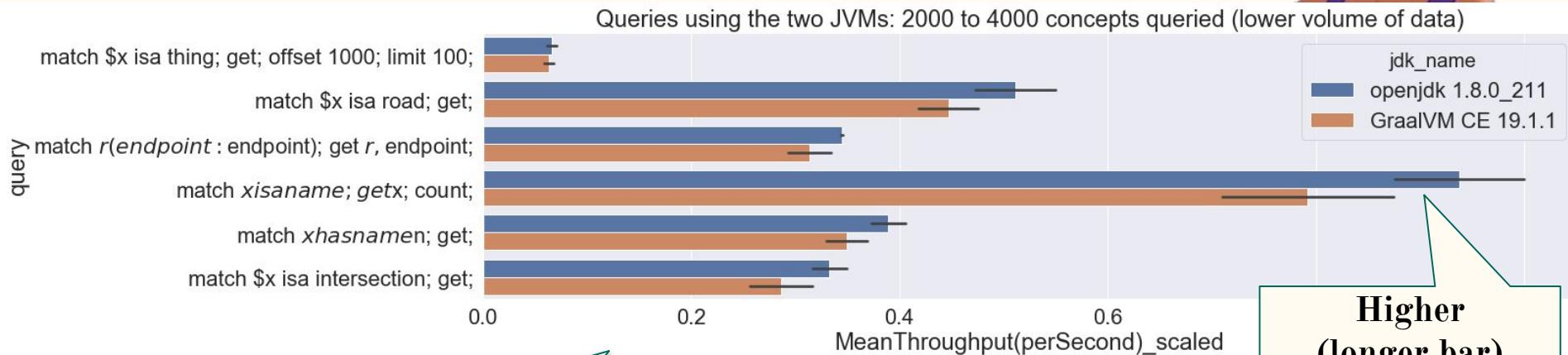
GraalVM:
quick wins!



```
$ ./measureTradVersusGraalVMStartupTime.sh
```

Running benchmark on Grakn server

Benchmark results (lower volume)



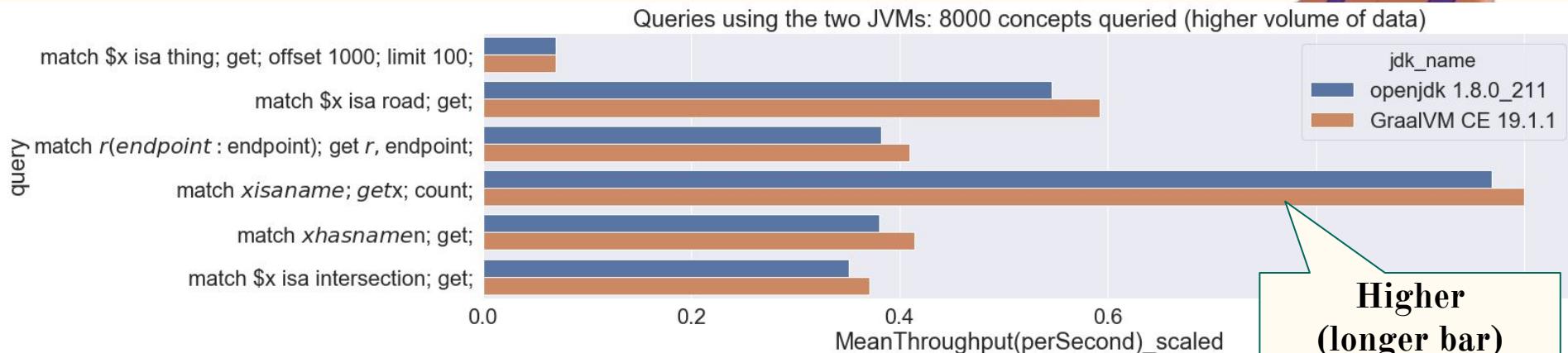
OpenJDK does better than GraalVM on these queries on lower volumes of data

Higher (longer bar) means better

Grakn 1.5.7
GraalVM CE
19.1.1

Benchmark results (higher volume)

And the latest
versions may do
even better!



GraalVM does *better* on
these queries with higher
volumes of data

Higher
(longer bar)
means better

Grakn 1.5.7
GraalVM CE
19.1.1

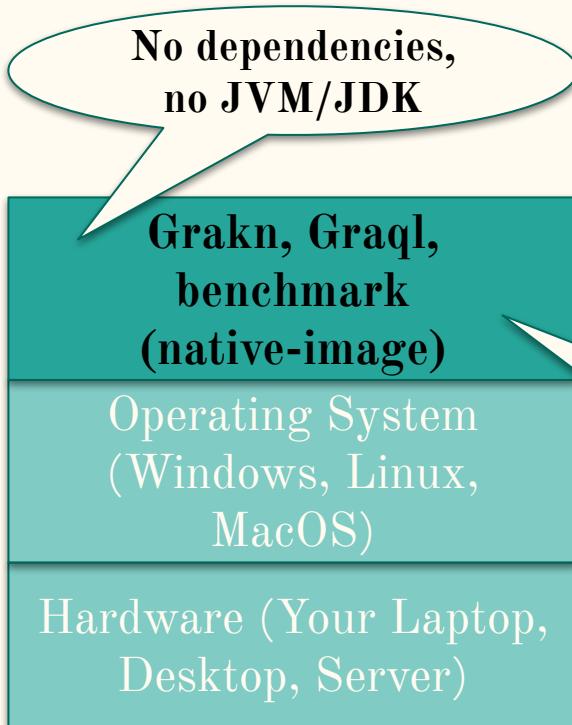
Grakn as native-image

GraalVM's native-image builder can convert ubjer JARs into *native binaries* (OS-specific)

native binaries have small-footprint, start quick, and have high runtime performance!

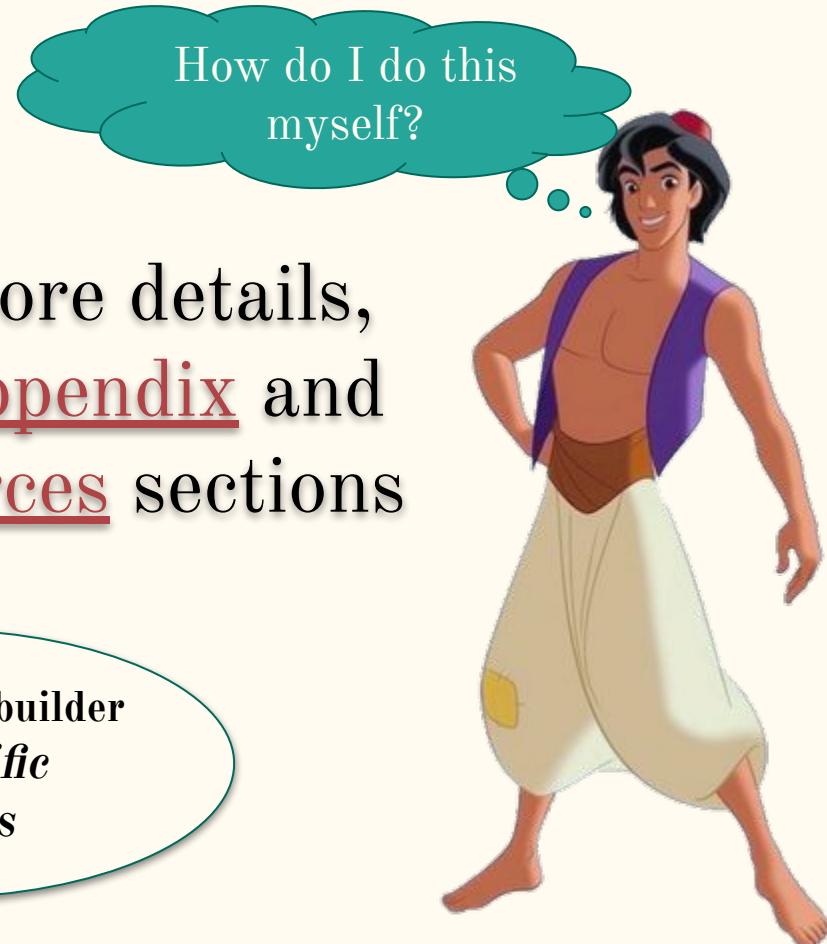


Grakn as native-image



For more details,
see Appendix and
Resources sections

native-image-builder
*OS-specific
binaries*



Quick recap of benefits

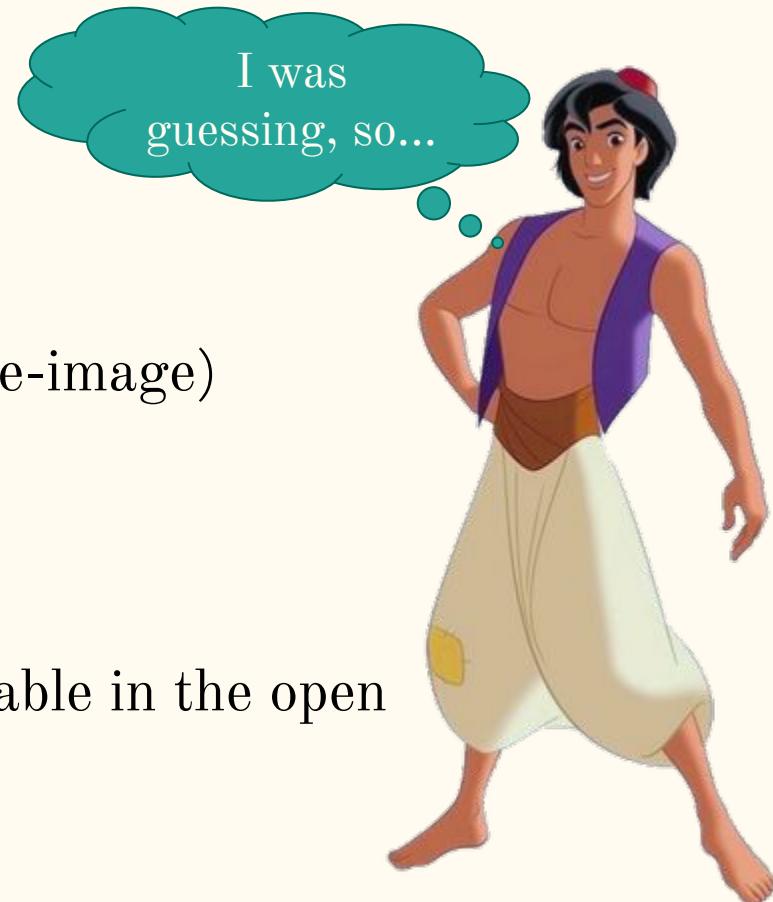
Benefits (JVM perspective)

- GraalVM works out-of-the-box as it is based on OpenJDK
- Primarily focussed on making the JVM performant
- In addition, allows creating OS-specific native-images
- Faster startup times
- Compact and dependency free images (native-image)
- Free Community version available
- Developed in the open (GitHub & Slack)
- Widely accepted and good community outreach



Benefits (Infra perspective)

- Fast and reliable response time
- Dependency free end-artifacts (native-image)
- Highly Portable
- Compact docker images
- Major platforms supported
- Regularly updated and updates available in the open



Quick disclaimer

These figures may vary in your case (i.e. environment)

Don't take them literally, make your own measurements!

Performance measurement is a skill: art and science!

Not making any claims: in reality, there are plenty of factors to consider!

Ok, ok, noted!



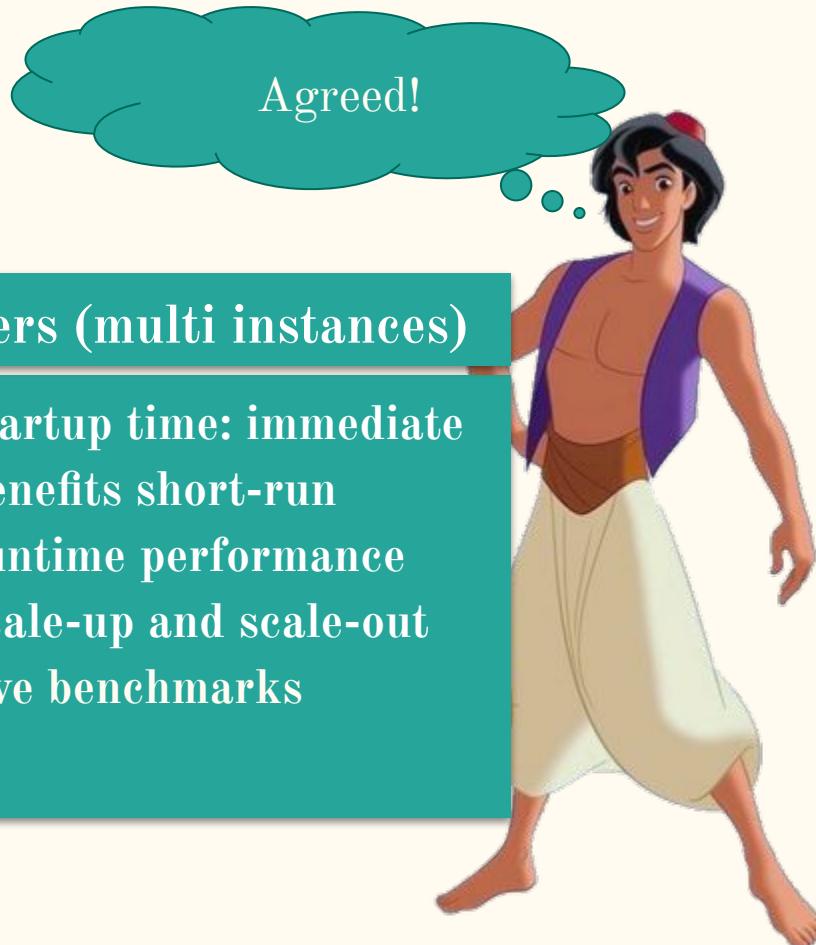
An instance or clusters

Single instance

- startup time: less in the short-run
- runtime performance
- prepare towards multi-instances
- live benchmarks

Clusters (multi instances)

- startup time: immediate benefits short-run
- runtime performance
- scale-up and scale-out
- live benchmarks



Being wrong isn't bad...

Be believe same!



It's times like these I
take support from
his quotes!

You Retweeted

Richard Feynman @ProfFeynman · 19h

Being wrong isn't a bad thing like they teach you in school. It is an opportunity to learn something.

A black and white photograph of Richard Feynman, a Nobel laureate in Physics, sitting at a chalkboard. He is smiling and pointing upwards with his right hand. The chalkboard behind him is covered with various mathematical equations and diagrams, including one that looks like a Feynman diagram. Below the photo, there are engagement metrics: 1.2K retweets and 4.1K likes.

@theNeomatrix369

Resources

Great community support:
please star, watch, fork, share
the repo.

How do I do all of these
things, you spoke about?



Code on GitHub

<https://bit.ly/grakn-graql-graalvm>

Open-source,
extendable

Ready-to-use Docker image on Docker Hub

<https://hub.docker.com/r/neomatrix369/grakn>

Docker pulls are
constantly growing,
Thank You!

@theNeomatrix369

Similar to Grakn's
docker image

More questions...

Yes I have more to ask,
how much more time do
you have?

Find them in
the Appendix section
and Resources section



Just about to leave...



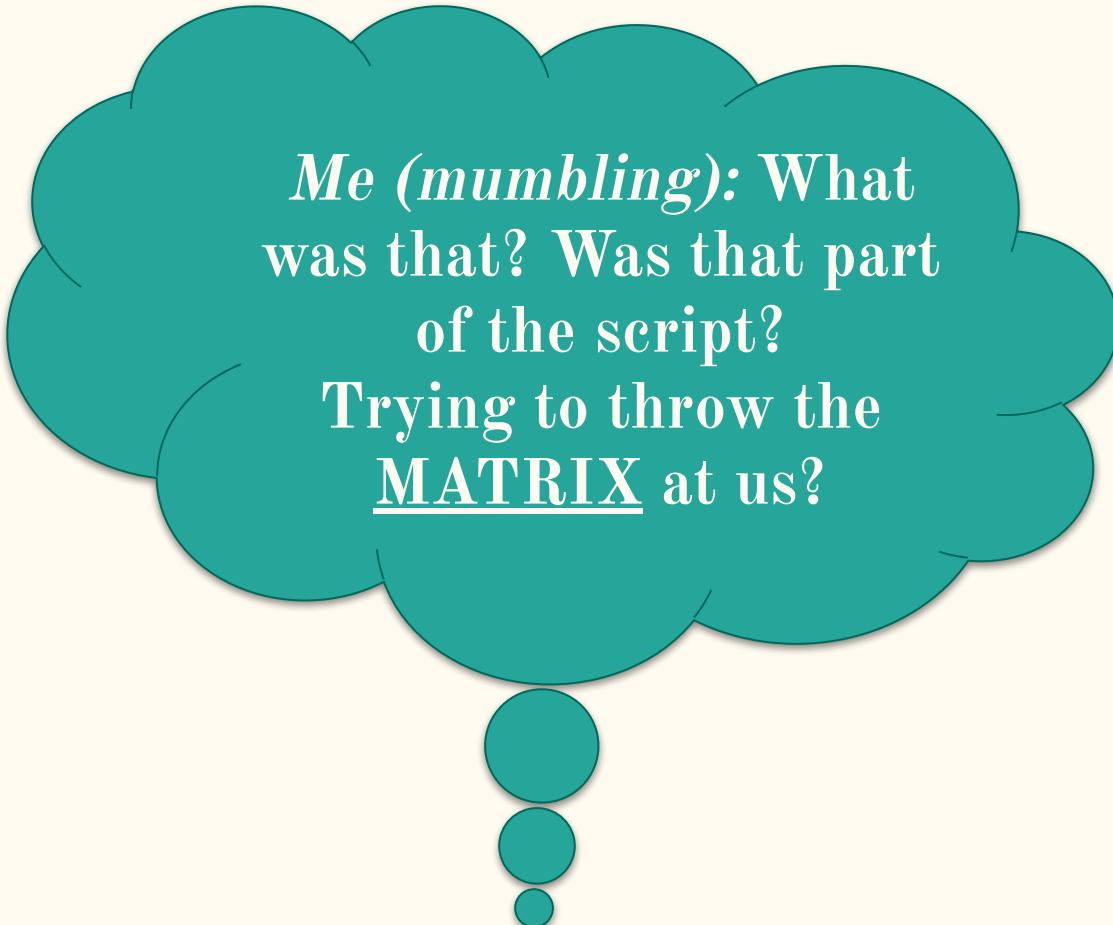
Hey, choose!
Which one?

Me: Weren't
you supposed
to ask this
right at the
start?

Me:
Umm,
PASS!

CHOOSE

Just after leaving...



Me (mumbling): What
was that? Was that part
of the script?
Trying to throw the
MATRIX at us?

Conversation with Jas

(meeting in the lobby)

Speaking Graql (with Jas)

Hi, again,
it's Jas

- Jas is all about Data Science, NLP, among other things
- She is keen on ubiquitous communications all across
- Her high-level goals with **Graql** are
 - Easy to learn and understand **Graql**
 - Write **Graql** for stakeholders of all levels
 - **Graql to English**
 - Translate communication from stakeholders of all levels to **Graql**
 - **English to Graql**



English to Graql

(graql-console-client with **GraqlBot**)

Speaking Graql: command-line



```
(host) $ ./run-python-in-docker.sh  
(container) $ pip3 install -r requirements.txt  
(container) $ python grakn_console_client.py
```

Run these from inside a
docker container

Speaking Graql: command-line

```
GraqlBot: Enter/paste your query in English or Graql.
```

```
And may the force be with us!
```

```
Type "exit" at the prompt to leave! "clear" to clear the screen.
```

```
GraqlBot: English or Graql >
```

Not like “Enter the dragon”, enter the **Graql!**

Great start!



Speaking Graql: entering queries

Choices, I like that!

GraqlBot: English or Graql >

Give me list of customers calling each other in London or Cambridge

GraqlBot: Not sure which one you meant! But we found others!

0 ---> Who are the customers who 1) have all called each other and 2) have all called person with phone number +48 894 777 5173 at least once?

Code: COMMON_CUSTOMERS_SINGLE_NUMBER | Confidence: 50-50 chance, 54%)

1 ---> Who are common contacts of customers with certain phone numbers

Code: COMMON_CUSTOMERS_MULTIPLE_NUMBERS | Confidence: 50-50 chance, 52%)

2 ---> Who people received call from customer of certain place aged over certain age also called by someone aged under certain age

Code: UNDER_20_PHONE_CALLS_LONDON? | Confidence: 50-50 chance, 49%)

3 ---> Get phone number of people received calls from customer aged customer potential person who made calls to another customer aged under certain age

Code: UNDER_20_PHONE_CALLS_LONDON? | Confidence: 50-50 chance, 45%)

4 ---> Who are the people aged under 20 who have received at least one phone call from a Cambridge customer aged over 50?

Code: OVER_50_PHONE_CALLS_CAMBRIDGE | Confidence: 50-50 chance, 45%)

GraqlBot: Which one of these did you mean, just type the q number?

GraqlBot: one of these: [0, 1, 2, 3, 4]



Filtering with saved queries!
Useful!

Speaking Graql: getting results

```
GraqlBot: Which one of these did you mean, just type the q number?  
GraqlBot: one of these: [0, 1, 2, 3, 4]
```

0

```
GraqlBot: Here's what the Graql query would look like if you typed it, neat isn't it?
```

```
match  
  $target isa person, has phone-number "+48 894 777 5173";  
  $company isa company, has name "Telecom";  
  $customer-a isa person, has phone-number $phone-number-a;  
  (customer: $customer-a, provider: $company) isa contract;  
  (caller: $customer-a, callee: $target) isa call;  
  $customer-b isa person, has phone-number $phone-number-b;  
  (customer: $customer-b, provider: $company) isa contract;  
  (caller: $customer-b, callee: $target) isa call;  
  (caller: $customer-a, callee: $customer-b) isa call;  
get $phone-number-a, $phone-number-b;
```

```
GraqlBot: Let me think, will take a moment, please be patient...
```

```
GraqlBot: The customers who have called the single number are  
['+81 308 988 7153', '+261 860 539 4754', '+62 107 530 7500', '+81 308 988 7153', '+261 860 539 4754', '+62 107 530 7500']
```

```
GraqlBot: And it took me 4.204568386077881 seconds (real-time) to execute this query.
```

```
GraqlBot: I could things faster if you like! I'm practising for the performance Olympics
```

Nicely done!

Could someone memorise that query?

I can,
more or less!!



Speaking Graql: automatic execution

```
GraqlBot: English or Graql >  
Since September which customers called
```

```
GraqlBot: Yay! We found it (at least we think we did)!
```

```
q0 ---> Since a date which customers called a person with phone number  
Code: CUSTOMERS_CALLED_SINCE | Confidence: Pretty likely, 82%
```

```
GraqlBot: Here's what the Graql query would look like if you typed it, neat isn't it?
```

```
match  
    $customer isa person, has phone-number $phone-number;  
    $company isa company, has name "Telecom";  
    ($customer: $customer, provider: $company) isa contract;  
    $target isa person, has phone-number "+86 921 547 9004";  
    ($caller: $customer, callee: $target) isa call, has started-at $started-at;  
    $min-date == 2018-09-10T00:00:00; $started-at > $min-date;  
get $phone-number;
```

```
GraqlBot: Let me think, will take a moment, please be patient...
```

```
GraqlBot: These are numbers of the customers who called +86 921 547 9004 since 2018-09-10T00:00:00  
['+263 498 495 0617', '+63 815 962 6097', '+81 308 988 7153', '+7 690 597 4443', '+54 398 559 0423', '+62 107 530  
7500', '+370 351 224 5176', '+81 746 154 2598']
```

```
GraqlBot: And it took me 3.4213337898254395 seconds (real-time) to execute this query.
```

Pretty
confident, hey



Quite close, safe
confidence
ratings!

Speaking Graql: similar query

GraqlBot: English or Graql >

Calls related to September 10th or 14th

GraqlBot: Not sure which one you meant! But we found others!

q0 ---> Get me the customers of company "Telecom" who called the target person with phone number +86 921 547 9004 from September 10th onwards.

Code: CUSTOMERS_CALLED_SINCE | Confidence: 50-50 chance, 55%

q1 ---> Get phone number of people received calls from customer aged customer potential person who made calls to another customer aged under certain age

Code: UNDER_20_PHONE_CALLS_LONDON? | Confidence: 50-50 chance, 49%

q2 ---> Get phone number of people received calls from customer aged customer potential person who made calls to another customer aged under certain age

Code: OVER_50_PHONE_CALLS_CAMBRIDGE | Confidence: 50-50 chance, 49%

q3 ---> Get me the phone number of people who have received calls from both customer with phone number +7 171 898 0853 and customer with phone number +370 351 224 5176.

Code: COMMON_CUSTOMERS_MULTIPLE_NUMBERS | Confidence: 50-50 chance, 49%

q4 ---> People aged under certain age received at least one phone call from a place customer from customer aged over certain age

Code: OVER_50_PHONE_CALLS_CAMBRIDGE | Confidence: 50-50 chance, 44%

GraqlBot: Which one of these did you mean, just type the q number?

GraqlBot: one of these: [0, 1, 2, 3, 4]



Speaking Graql: results from cache

GraqlBot: Which one of these did you mean, just type the q number?

GraqlBot: one of these: [0, 1, 2, 3, 4]

0

GraqlBot: These are numbers of the customers who called +86 921 547 9004 since 2018-09-10T00:00:00
['+263 498 495 0617', '+63 815 962 6097', '+81 308 988 7153', '+7 690 597 4443', '+54 398 559 0423', '+62 107 530 7500', '+370 351 224 5176', '+81 746 154 2598']

GraqlBot: And it took me 2.86102294921875e-06 seconds (cache) to execute this query.

GraqlBot: Thats faster than Usain Bolt

GraqlBot: The above is based on your original input: 'Calls related to September 12th or 14th'

0.00000286
seconds

Cached actions are usually faster
(maybe less accurate)



Speaking Graql: manually graql-ing

And may the force be with us!

Type "exit" at the prompt to leave! "clear" to clear the screen.

GraqlBot: English or Graql >

```
graql: match $customer isa person, has age < 20; $company isa company, has name "Telecom"; (customer: $customer, provider: $company) isa contract; (caller: $customer, callee: $anyone) isa call, has duration $duration; get $duration; mean $duration;
```

GraqlBot: Wow, that's a great change, not many do that these days ;)

GraqlBot: Happy to execute it for you, if you think you have your graql-foo down

GraqlBot: Nice effort, looks like a well crafted query!

```
match $customer isa person, has age < 20; $company isa company, has name "Telecom"; (customer: $customer, provider: $company) isa contract; (caller: $customer, callee: $anyone) isa call, has duration $duration; get $duration; mean $duration;
```

GraqlBot: Let me think, will take a moment, please be patient (talking to Highlander Grakn Server)...

GraqlBot: Here's the output to your hand-written query: 1242.7714285714285

GraqlBot: And it took me 1.901900291442871 seconds (real-time) to execute this query.

GraqlBot: Even though it's been a long day, and I'm a bit lazy today!

```
GraqlBot: The above is based on your original input: 'graql: match $customer isa person, has age < 20; $company isa company, has name "Telecom"; (customer: $customer, provider: $company) isa contract; (caller: $customer, callee: $anyone) isa call, has duration $duration; get $duration; mean $duration;'
```

GraqlBot: English or Graql >

Hey! You can do
that as well!



User typed in *graql*
query

Me: Yeah,
old school
still works!

Speaking Graql: any query

GraqlBot: Enter/paste your query in English or Graql, exit to leave the prompt! (Let the force be with us!)

GraqlBot: English or Graql >

aSKLDjLAKSdjLAKSd

GraqlBot: Not sure what you meant by that one, not the end of the world. We can try again.

GraqlBot: English or Graql >

askdjaslkdkjaskldjaskldj

GraqlBot: Nice try, but we could find nothing! Do you want to try another query?

GraqlBot: English or Graql >

djfhadskjfjhdkjf

GraqlBot: Don't give up on me just cause I don't follow you. Keep trying till we perfect it

GraqlBot: English or Graql >

v,dms.,vmads.,v

GraqlBot: Not sure what you meant by that one, not the end of the world. We can try again.

GraqlBot: English or Graql >

Hmm, let
me think!

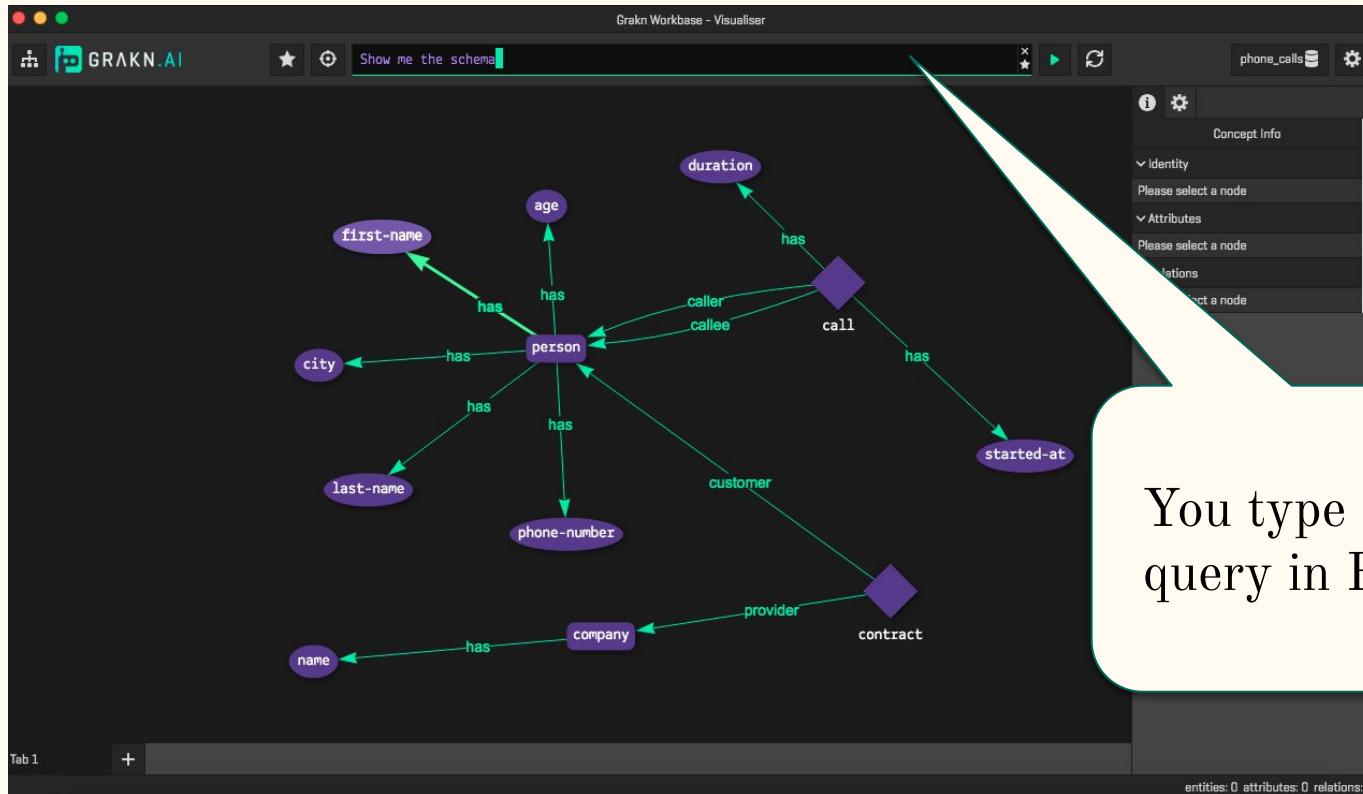
Keep
trying!
Everyone
should get
multiple
chances.
Who else
thinks so?



Workbase

That looks easy
to read

Speaking Graql: type in English



You type in your
query in English



Now! best of
both worlds, hey!

Speaking Graql: respond in English

Graql Workbase - Visualiser

GRAKN.AI

Show me the schema

```
GraqlBot: This is how your Graql query looks like, neat hey!
match $x sub thing;get;
```

The screenshot shows the Graql Workbase Visualiser interface. On the left, there's a code editor with a purple background containing a Graql query. To the right is a graph visualization of the schema. A large teal arrow points from the text area down towards the graph. Below the graph, another text area displays the generated Graql query.

Show me the schema

```
GraqlBot: This is how your Graql query looks like, neat hey!
match $x sub thing;get;
```

entities: 0 attributes: 0 relations: 0



You type in your query in English,
and GraqlBot does the rest!

Graql to English

(Decomposing a graql Query)

graql query

Query looks good,
like the style and
formatting 😊

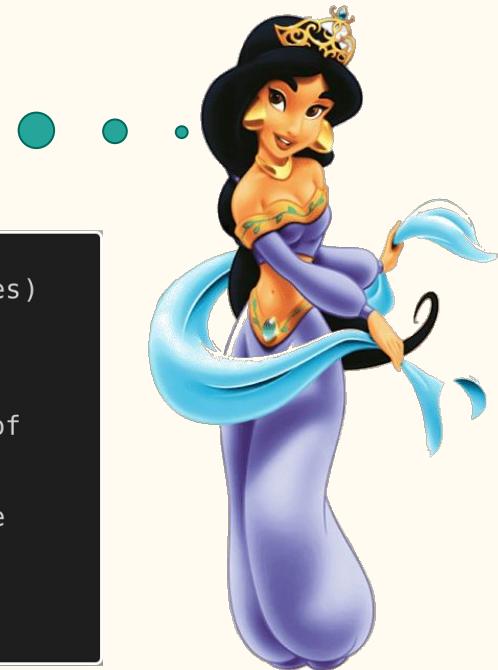
```
1 match
2   $customer isa person, has phone-number $phone-number;
3   $company isa company, has name "Telecom";
4   (customer: $customer, provider: $company) isa contract;
5   $target isa person, has phone-number "+86 921 547 9004";
6   (caller: $customer, callee: $target) isa call, has started-at $started-at;
7   $min-date == 2018-09-10T00:00:00; $started-at > $min-date;
8 get $phone-number;
```



In English: literal translation

- ```
1 - Find customers of type person, with attribute phone number populated (valid entries)
2 - Find a company whose attribute name is Telecom
3 - Find a contract relationship between a customer and company
4 - Find a target (customer) of entity type person and with a phone number attribute of
 value "+86 921 547 9004"
5 - Find a call relationship between a customer and a target (also customer) where the
 call attribute started-at > a given minimum date of 2018-09-10T00:00:00
6 - Find the phone number of the target
```

I would have sort of  
done the same. 😊



# In English: some simplifications

Looks a bit more  
compact, without the  
unneeded details.



- 1 - Find a company by the name Telecom
- 2 - And customers who have a contract with this company
- 3 - And find the customer with the telephone number "+86 921 547 9004"
- 4 - Find all the calls made by this customer to others on or after the date 10-9-2018 starting mid-night (or September 10th 2018 starting mid-night)
- 5 - Gather and return the phone numbers of all these people who were called

# In English: further simplifications

Looks like we are getting close to its crux and essence!



- 1 - Find the customer from the company Telecom with the telephone number "+86 921 547 9004"
- 2 - Get the numbers of all the calls made by this customer on or after the date 10-9-2018 mid-night (or September 10th 2018 mid-night)



In English: amenable-to-the-eyes

Aah really, that sounds more  
like how our stakeholders talk  
to us!



Since September 10th, which customers called  
this number +86 921 547 9004?

# In English: text summarisation

It appears to be more about text summarisation  
(look for notebooks on text summarisations)

Groovy,  
notebooks! I love them!



Me: not Groovy, but Python notebooks with a bit of Markdown text 😊

# Quick recap

The two-way translation process

There is no “one hop” solution,  
it is an iterative process!

Made up of layers!

The two-way translation process

Let me try to explain

# Graql to English translation

```
1 match
2
3 1 - Find customers of type person, with attribute phone number populated
4
5 2
6 1 - Find a company by the name Telecom
7
8 g
9
10 5 1 - Find the customer from the company Telecom with the telephone number
11
12 6 2
13 1
14 1 - Since September 10th, which customers called
15 2
16 this number +86 921 547 9004?
```

We gain and  
lose clarity,  
intent, i.e.  
necessary  
details!

# Towards a smooth two-way speech

Two-way: you mean, round-trip!

English-to-Graql

Graql-to-English

From it's creators!

Graql

And the community

Alignments



Adjustments

Abstractions

We are not far off!

Adjustments



Alignments

English

And the community

From it's users!

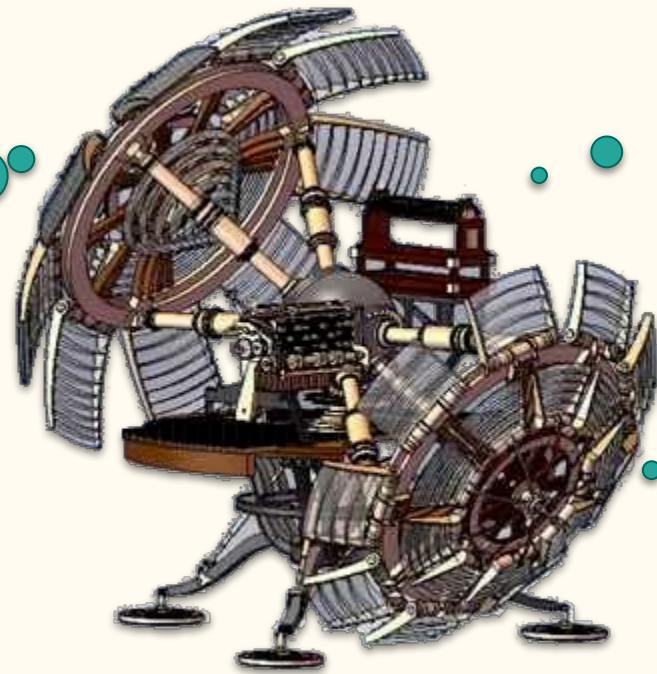


# A little while later

(After finishing conversation with Jas)

# Where did “Jas” find the blog?

Plan to  
write one  
but haven't  
written one  
yet!



Does Jas have  
a time  
machine?

Is Jas from  
the future?  
#irony

# Summary

- Grakn
  - Improve the runtime performance of modern day programs running on the JVM. Solution to do that is available and easy to use! Contributions are welcome!
  - **Grakn** is versatile and easy to adapt!
- Graql
  - Beauty of **Graql** you can perform two-way translations (**E-2-G, G-2-E**)
  - We can use simple **pattern recognition** to deliver NLP-like solutions without doing full-fledge model building
- Al and Jas are happy users!

# Closing notes

# Community support



GraknLabs  
heartily welcomes  
contributors and  
collaborators



I'm looking for  
collaborators,  
please reach out!



Big “thank you”  
to the community  
for your long  
lasting support!

# Resources

- These Slides: <https://bit.ly/grakn-graql-graalvm-slides>
- Github: <https://bit.ly/grakn-graql-graalvm>
- [Grakn project](#) (referred in this presentation) [highly recommended]
- [Grakn Docker Hub Image](#)
- [Awesome AI/ML/DL](#) | [Graphs](#) | [Graph Databases](#)
- Follow [GraknLab](#)'s performance initiative(s) on GitHub: [[1](#)] [[2](#)]
  - the [benchmark](#) project mentioned in this presentation | [Paper on VM benchmarking](#)
- **Graql:** English-to-Graql, Graql-to-English
  - Discussion on GitHub: [[1](#)][[2](#)][[3](#)]
  - Follow [mauna.ai](#)
- [GraalVM](#) | [GitHub](#) | [Awesome Graal](#) | [Script to build native-image of Grakn](#)

And everyone  
else at  
GraknLabs

# Thank You, again!

- Grakn Cosmos 2020, its organisers (Daniel Crowe)
- Good folks at GraknLabs (Haikal, Tomas, Joshua...)
- Developers from the various developer communities I have been at over the years
- You for being here, sparing your time, and entrusting it with me for this session



@theNeomatrix369



# Contact and keep in touch

- twitter: [@theNeomatrix369](#)
- medium: <https://medium.com/@neomatrix369>
- github: <https://github.com/neomatrix369/>
- linkedin: <https://www.linkedin.com/in/mani-sarkar/>
- youtube: [channel | playlists](#)
- about me: <https://neomatrix369.wordpress.com/about>

# Q & A

Movie reference:  
Aladdin and the  
Magic Lamp?

Al = Aladdin?

Jas = Jasmine?

Does Jas have a time  
machine?

Red pill, blue ill,  
movie reference?  
Matrix?

Is Al from the  
Matrix?

Being fictional  
stars: Al and Jas  
may not be able to  
answer questions!

# Appendix

# Al: questions, discussions, ideas

Al: I have more questions!

It's not only about which JDK is better or the best

Benchmark and log metrics

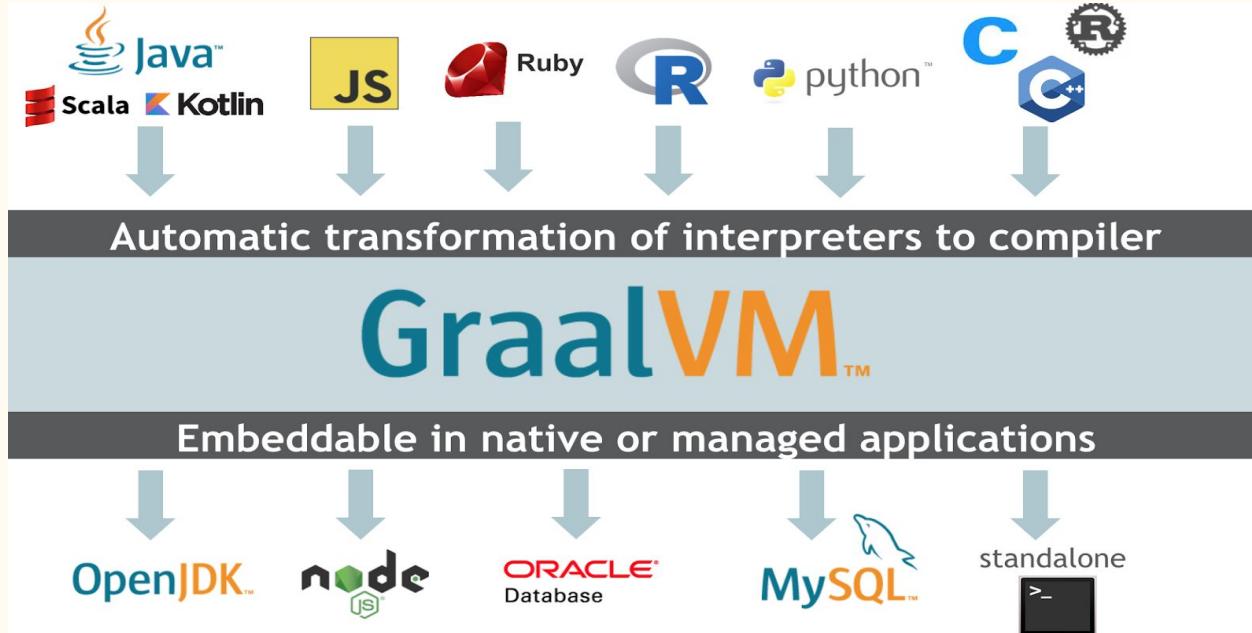
Cluster of servers behind a load balance, running different JDKs

Have a setup in place to be able to run Grakn on the best of breed infra for optimum performance and throughput

Different JDKs for different end-goals

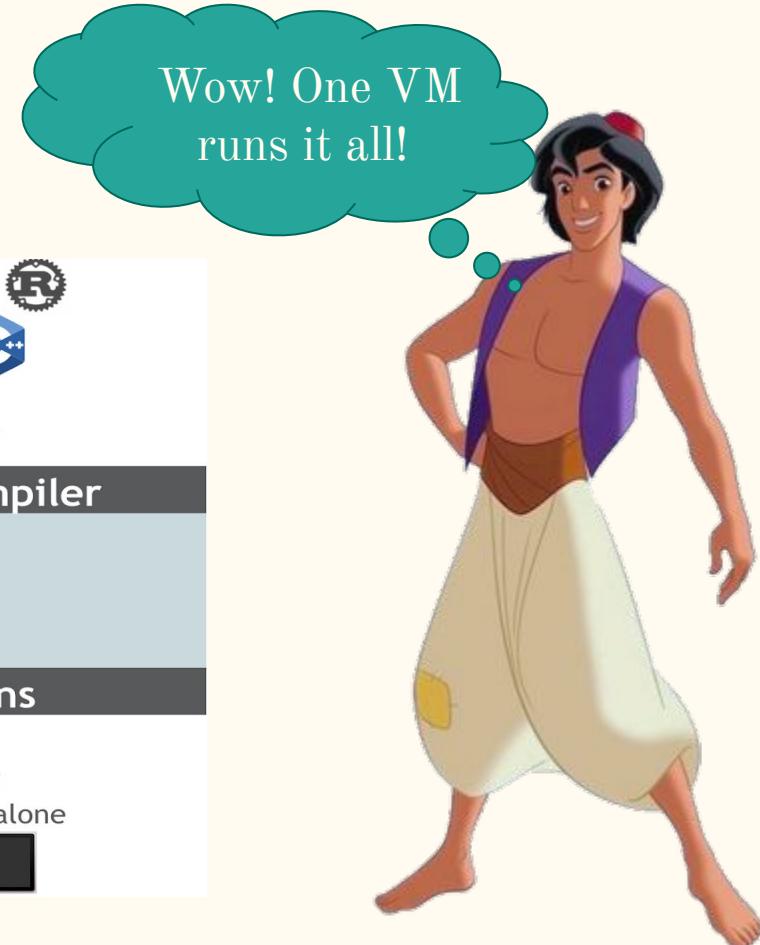
Measure startup and runtime performances between GraalVM and another JDKs

# GraalVM architecture

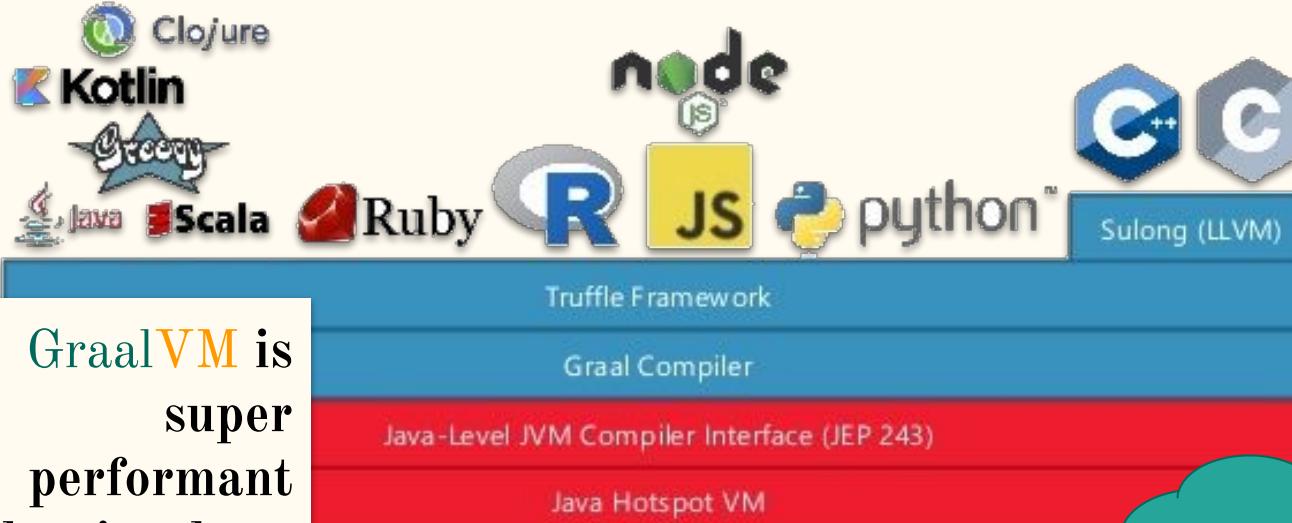


High-level

@theNeomatrix369



# GraalVM Architecture



GraalVM is  
super  
performant  
despite these  
layers of  
abstractions

@theNeomatrix369

I like to know  
more, later on!



# GraalVM native-image

- Convert uber JAR files into native-binary
- OS-specific native binaries
- Zero or very little dependencies needed (OS-specific)
- Quick startup
- Small footprint
- Ideal for docker images
- Performant at runtime
- Create highly performant images using profile guided process (available only via GraalVM EE version)
- See [script to build native-image of Grakn](#)

