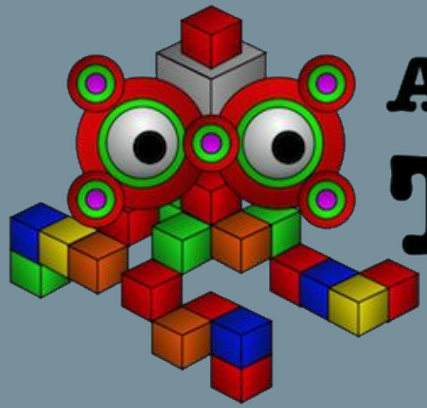
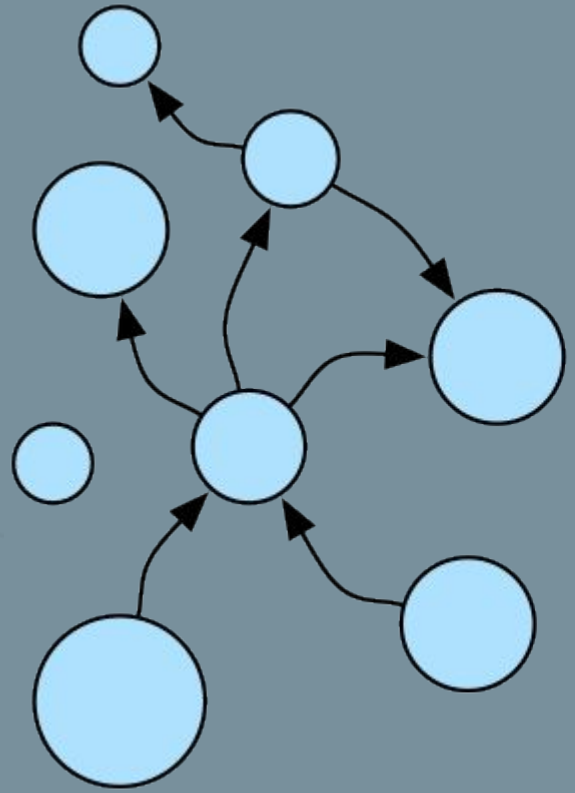


Apache TinkerPop

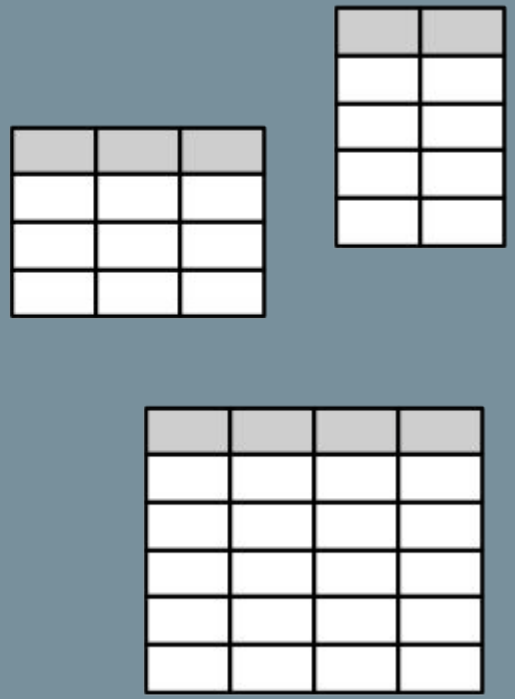
Gremlin

$$G = (V, E)$$

Graph querying and traversal



VS



Graph querying and traversal

OrientDB

TITAN

blazegraph™

OLTP

Gremlin

$G = (V, E)$

OLAP

APACHE GIRAPH

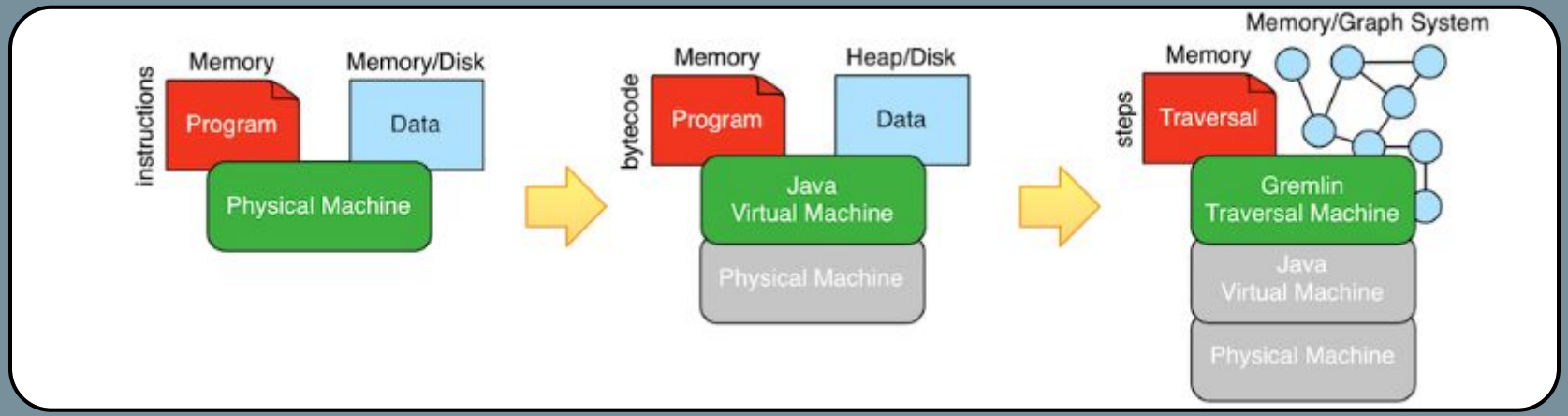
hadoop

Spark™

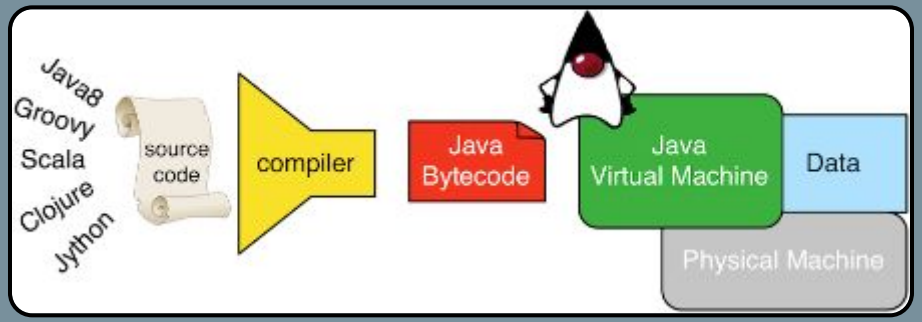
Graph Databases

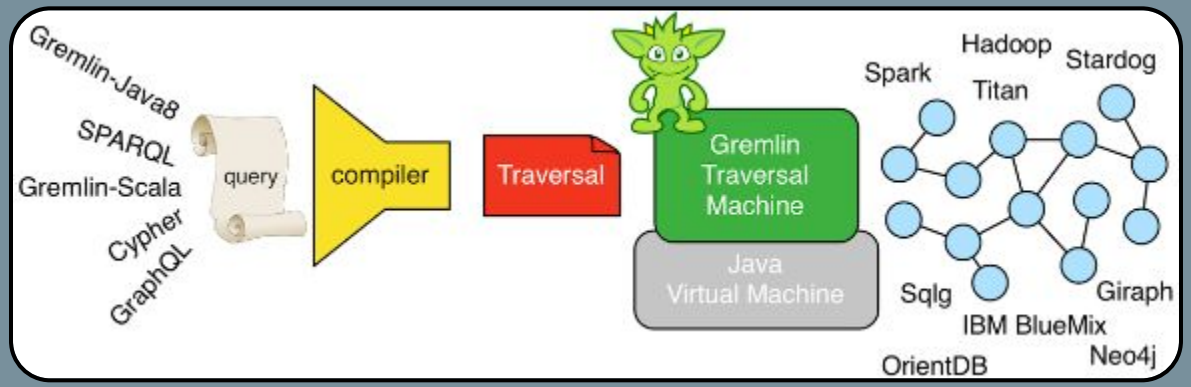
Graph Processors

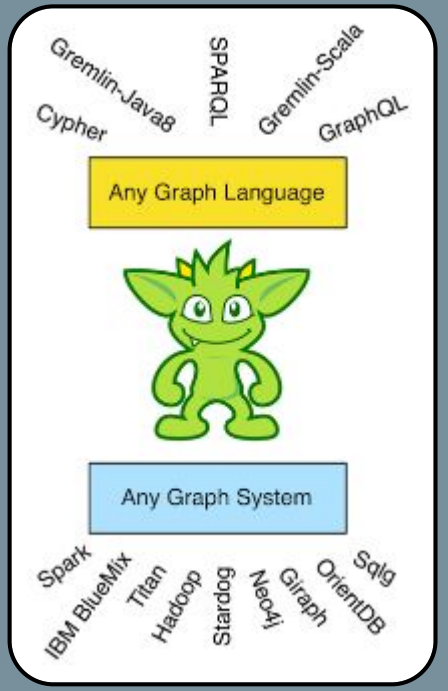
Language agnostic: Write once, run everywhere (so long as it's Java)



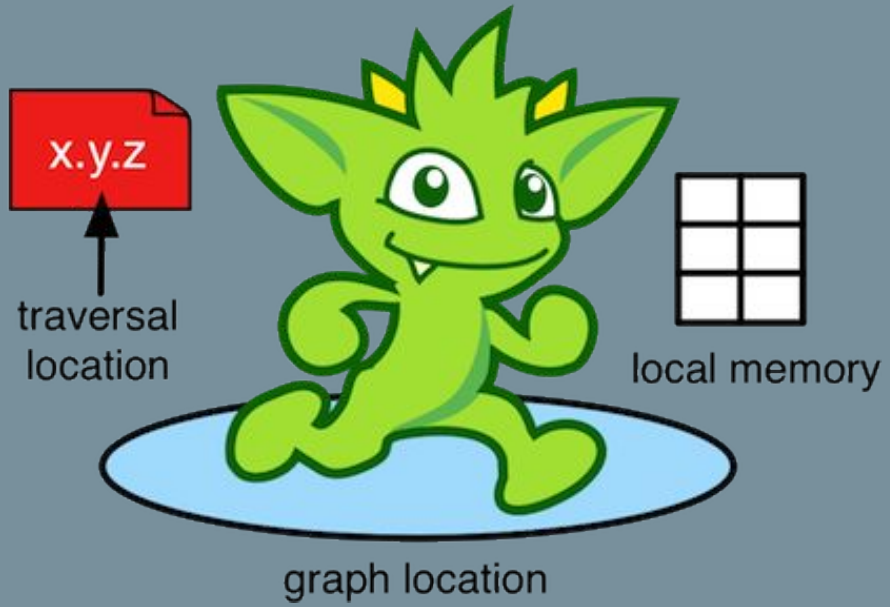
Language agnostic: Write once, run everywhere (so long as it's Java)



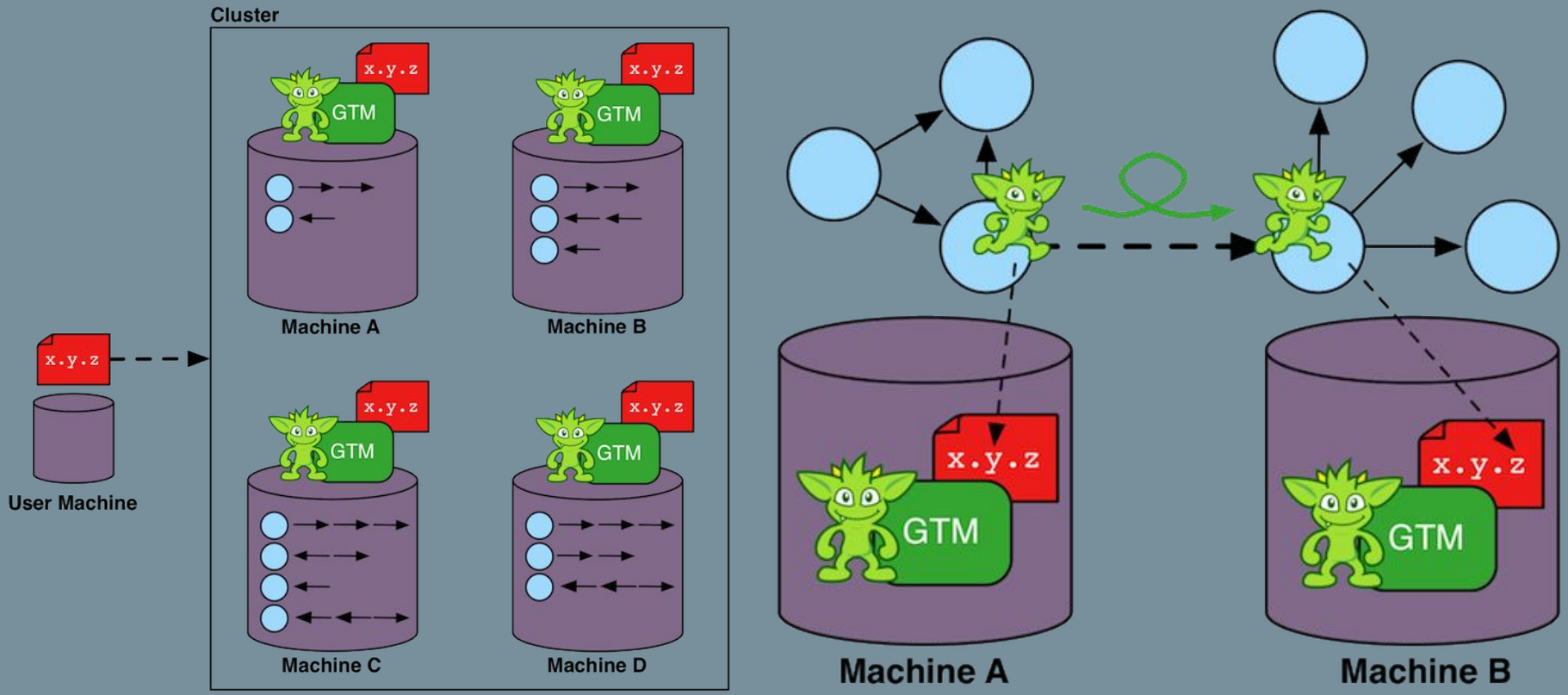




Gremlin traverser



Operation in a distributed system



Gremlin syntax: a couple simple examples

```
g.V().has("name","gremlin").
  out("knows").
  out("knows").
  values("name")
```

What are the names of Gremlin's friends' friends?

1. Get the vertex with name "gremlin."
2. Traverse to the people that Gremlin knows.
3. Traverse to the people those people know.
4. Get those people's names.

```
g.V().match(
  as("a").out("knows").as("b"),
  as("a").out("created").as("c"),
  as("b").out("created").as("c"),
  as("c").in("created").count().is(2)).
    select("c").by("name")
```

What are the names of the projects created by two friends?

1. ...there exists some "a" who knows "b".
2. ...there exists some "a" who created "c".
3. ...there exists some "b" who created "c".
4. ...there exists some "c" created by 2 people.
5. Get the name of all matching "c" projects.

Gremlin syntax: a couple complex examples

```
g.V().has("name","gremlin").
  repeat(in("manages")).
    until(has("title","ceo")).
  path().by("name")
```

Get the managers from Gremlin to the CEO in the hiearchy.

1. Get the vertex with the name "gremlin."
2. Traverse up the management chain...
3. ...until a person with the title of CEO is reached.
4. Get name of the managers in the path traversed.

```
g.V().has("name","gremlin").as("a").
  out("created").in("created").
    where(neq("a")).
  groupCount().by("title")
```

Get the distribution of titles amongst Gremlin's collaborators.

1. Get the vertex with the name "gremlin" and label it "a."
2. Get Gremlin's created projects and then who created them...
3. ...that are not Gremlin.
4. Group count those collaborators by their titles.

Gremlin syntax: a couple more complex examples

```
g.V().has("name","gremlin").
  out("bought").aggregate("stash").
  in("bought").out("bought").
    where(not(within("stash"))).
  groupCount().order(local).by(values,desc)
```

Get a ranked list of relevant products for Gremlin to purchase.

1. Get the vertex with the name "gremlin."
2. Get the products Gremlin has purchased and save as "stash."
3. Who else bought those products and what else did they buy...
4. ...that Gremlin has not already purchased.
5. Group count the products and order by their relevance.

```
g.V().hasLabel("person").
  pageRank().
    by("friendRank").
    by(outE("knows")).
  order().by("friendRank",desc).
  limit(10)
```

Get the 10 most central people in the knows-graph.

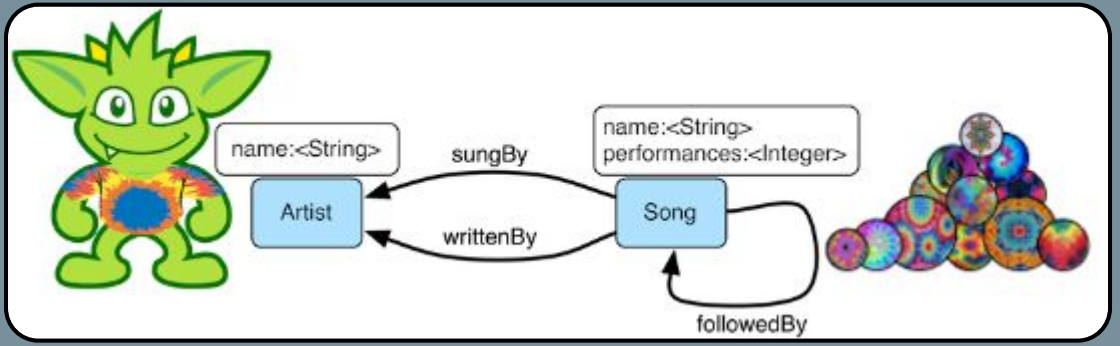
1. Get all people vertices.
2. Calculate their PageRank using knows-edges.
3. Order the people by their friendRank score.
4. Get the top 10 ranked people.

Jerry Garcia

Which song writers wrote songs that were sung by Jerry Garcia and performed by the Grateful Dead more than 300 times?

This query is expressed in Gremlin-Java8 as:

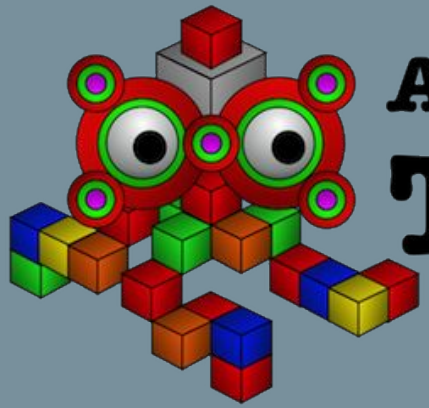
```
g.V().match(
  as("song").out("sungBy").as("artist"),
  as("song").out("writtenBy").as("writer"),
  as("artist").has("name","Garcia"),
    where(as("song").values("performances").is(gt(300)))
).select("song","writer").by("name")
```



Jerry Garcia

```
$ bin/gremlin.sh

         \,,,/
         (o o)
-----oOOo-(3)-oOOo-----
plugin activated: aurelius.titan
plugin activated: tinkerpop.server
plugin activated: tinkerpop.utilities
plugin activated: tinkerpop.hadoop
plugin activated: tinkerpop.tinkergraph
gremlin> :install com.datastax sparql-gremlin 0.1
==>Loaded: [com.datastax, sparql-gremlin, 0.1]
gremlin> :plugin use datastax.sparql
==>datastax.sparql activated
gremlin> graph = TitanFactory.open('conf/titan-cassandra.properties')
==>standardtitangraph[cassandrathrift:[127.0.0.1]]
gremlin> :remote connect datastax.sparql graph
gremlin> query = """
  SELECT ?songName ?writerName WHERE {
    ?song e:sungBy ?artist .
    ?song e:writtenBy ?writer .
    ?song v:name ?songName .
    ?writer v:name ?writerName .
    ?artist v:name "Garcia" .
    ?song v:performances ?performances .
      FILTER (?performances > 300)
  }
"""
gremlin> :> @query
==>[songName:BERTHA, writerName:Hunter]
==>[songName:TENNESSEE JED, writerName:Hunter]
==>[songName:BROWN EYED WOMEN, writerName:Hunter]
==>[songName:CHINA CAT SUNFLOWER, writerName:Hunter]
==>[songName:CASEY JONES, writerName:Hunter]
==>[songName:BLACK PETER, writerName:Hunter]
==>[songName:RAMBLE ON ROSE, writerName:Hunter]
==>[songName:WHARF RAT, writerName:Hunter]
==>[songName:LADY WITH A FAN, writerName:Hunter]
==>[songName:HES GONE, writerName:Hunter]
==>[songName:LOSER, writerName:Hunter]
==>[songName:DEAL, writerName:Hunter]
==>[songName:SUGAREE, writerName:Hunter]
==>[songName:DONT EASE ME IN, writerName:Traditional]
==>[songName:UNCLE JOHNS BAND, writerName:Hunter]
==>[songName:SCARLET BEGONIAS, writerName:Hunter]
==>[songName:EYES OF THE WORLD, writerName:Hunter]
==>[songName:US BLUES, writerName:Hunter]
==>[songName:TERRAPIN STATION, writerName:Hunter]
==>[songName:STELLA BLUE, writerName:Hunter]
gremlin>
```



Apache TinkerPop

Gremlin

$G = (V, E)$