Introduction to Gremlin

Online Course - July 21, 2017

Section 1: Introduction

What are Graphs and why should I care about TinkerPop?

Topics to cover:

- What is a Property Graph?
- What is Apache TinkerPop?
- Overview of TinkerPop Building Blocks
- Introduction to the Gremlin Console

What is a Property Graph?

What is TinkerPop?

Overview of TinkerPop Building Blocks

Exercise 1: Apache TinkerPop Concepts

a. What is a Property Graph?
b. What organization sponsors TinkerPop?
c. What type of thing is a Graph in TinkerPop?
d. What type of thing is a Traversal in TinkerPop?
e. What must you **always** do with the result of a TinkerPop Traversal?

Gremlin Console

To start your Gremlin Console:

1. Open a Terminal / CMD prompt for Docker commands
2. Download or git clone this repo: https://github.com/experoinc/gremlin-lang-intro
3. Build the Docker image: **`docker build -t gremlin-lang-intro .`**
4. Start a Docker container: **`docker run -it gremlin-lang-intro`**

Gremlin Console Commands

```
gremlin> :help
For information about Groovy, visit:
    http://groovy-lang.org

Available commands:
  :help       (:h  ) Display this help message
  ?           (:?  ) Alias to: :help
  :exit       (:x  ) Exit the shell
  :quit       (:q  ) Alias to: :exit
  import      (:i  ) Import a class into the namespace
  :display    (:d  ) Display the current buffer
  :clear      (:c  ) Clear the buffer and reset the prompt counter
  :show       (:S  ) Show variables, classes or imports
  :inspect    (:n  ) Inspect a variable or the last result with the ...
  :purge      (:p  ) Purge variables, classes, imports or preferences
  :edit       (:e  ) Edit the current buffer
  :load       (:l  ) Load a file or URL into the buffer
  .           (:.  ) Alias to: :load
  :save       (:s  ) Save the current buffer to a file
  :record     (:r  ) Record the current session to a file
  :history    (:H  ) Display, manage and recall edit-line history
  :alias      (:a  ) Create an alias
  :register   (:rc ) Register a new command with the shell
  :doc        (:D  ) Open a browser window displaying the doc for ...
  :set        (:=  ) Set (or list) preferences
  :uninstall  (:-  ) Uninstall a Maven library and its dependencies ...
  :install    (:+  ) Install a Maven library and its dependencies ...
  :plugin     (:pin) Manage plugins for the Console
  :remote     (:rem) Define a remote connection
  :submit     (:>  ) Send a Gremlin script to Gremlin Server
For help on a specific command type:
    :help command

gremlin> :remote
==>Remote - Gremlin Server - [localhost/127.0.0.1:8182]-[uuid]
gremlin> g
==>graphtraversalsource[tinkergraph[vertices:1491 edges:5324], standard]
gremlin> :x
```

You will need to restart your Docker container after exiting the console.

Gremlin Console Commands

```
gremlin> graph
==>tinkergraph[vertices:1491 edges:5324]
gremlin> g
==>graphtraversalsource[tinkergraph[vertices:1491 edges:5324], standard]
gremlin> g.V().count()
==>1491
gremlin> g.E().count()
==>5324
```

Exercise 2: Gremlin Server

Use the Gremlin Server log entries to answer the following. Restart the Docker container if necessary.

a. What type of file is used to Configure the Gremlin Server? What directory is it in?
b. How many different `Graphs` are loaded by the Graph Manager? What are the `Graph` variable names?
c. What `SciptEngine` is loaded?
d. How many `GraphTraversalSources` are bound at startup? What are the `GraphTraversalSource` variable names?
e. What port does the Gremlin Server use?

Exercise 3: Gremlin Console

Answer the following questions and/or perform the following actions:

a. How many plugins are activated when the Gremlin Console started?
b. What key operation does the Gremlin Console automatically do for you?
c. How do you get help for Gremlin Console commands?
d. What Gremlin Server does the Gremlin Console have a remote connection to?
e. What do you think the UUID in the :remote response is for?
f. Enter each Graph variable name (e.g. classic, modern,... from Exercise 2b) one at a time and note the size of the graph. What is the largest graph? Which is the smallest?
g. Enter each TraversalSource name one at a time and note the type of Graph they operate against.

Introduction to Gremlin

Online Course - July 21, 2017

Section 2: Basic Traversals

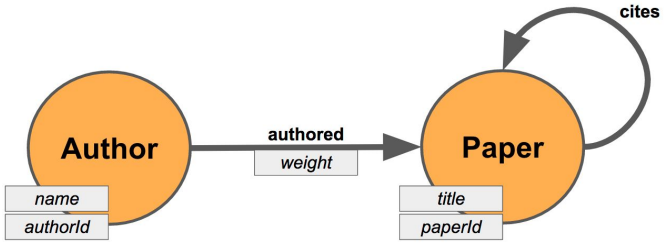
The R. you've been dreaming of

Topics to cover:

- Finding Vertices & Edges
- Traversing Edges
- Filtering

Best Practices & Defensive Coding

The Citations Data Set



Finding Vertices

g.V()

```
gremlin> g.V().limit(5)
==>v[0]
==>v[2049]
==>v[4098]
==>v[3]
==>v[2052]
```

g.V(#)

```
gremlin> g.V(594)
==>v[594]
```

g.V().hasId(#)

```
gremlin> g.V().hasId(594)
==>v[594]
```

g.V().has(key, value)

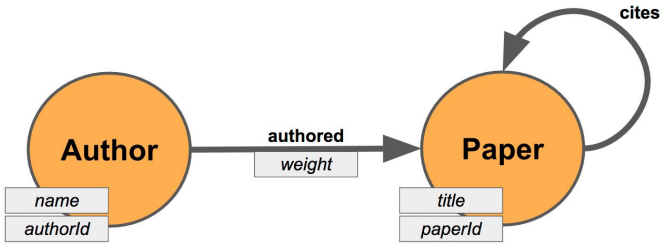
```
gremlin> g.V().has('name','H Zhi')
==>v[2049]
```

g.V().has(label,key,value)

```
gremlin> g.V().has('Author','name','H Zhi')
==>v[2049]
```

g.V().has(Type,label|id)

```
gremlin>
g.V().has(label,'Author').limit(5)
==>v[0]
==>v[2049]
==>v[3]
==>v[2052]
==>v[6]
gremlin> g.V().has(id,2049)
==>v[2049]
```



g.V().properties(key).hasValue(value)

```
gremlin> g.V().properties('name').hasValue('H Zhi')
==>vp[name->H Zhi]
```

Finding Edges

g.E()

```
gremlin> g.E().limit(5)
==>e[8192][3726-cites->2757]
==>e[8193][3726-cites->2964]
==>e[8194][3726-cites->2721]
==>e[8195][3726-cites->2838]
==>e[8196][3726-cites->2805]
```

g.E().has(Type,value)

```
gremlin> g.E().has(label,'authored').limit(5)
==>e[4473][0-authored->2418]
==>e[4474][3-authored->2421]
==>e[4475][6-authored->2421]
==>e[4476][9-authored->2424]
==>e[4477][12-authored->2424]
```

g.E().hasLabel(label)

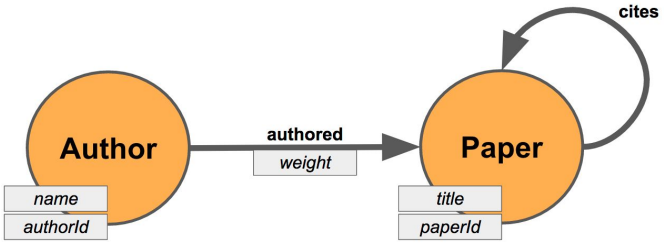
```
gremlin> g.E().hasLabel('authored').limit(5)
==>e[4473][0-authored->2418]
==>e[4474][3-authored->2421]
==>e[4475][6-authored->2421]
==>e[4476][9-authored->2424]
==>e[4477][12-authored->2424]
```

Exercise 4: Finding Vertices and Edges

Answer the following questions. Use the Citations graph for writing and testing traversals.

a. What two reasons are there for always using a Traversal object instead of a Graph one?
b. What does the Gremlin Console automatically do to the traversal results for you?
c. What type of Element does the Graph Step g.V() return?
d. What type of Element does the Graph Step g.E() return?
e. What types of Elements can we filter on?
f. Return five Paper vertices from the graph.
g. Return five cites edges from the graph.
h. Find the ID of the Person vertex with the name: “R Li”
i. Find the vertex with the internal ID “3” using two different traversals.

Getting Properties



g.V().values()

```
gremlin> g.V(594).values()
==>A Ambainis
==>199
```

g.V().values('name')

```
gremlin> g.V(594).values('name')
==>A Ambainis
```

g.V().valueMap()

```
gremlin> g.V(594).valueMap()
==>{name=[A Ambainis], authorId=[199]}
```

g.V().valueMap('name')

```
gremlin> g.V(594).valueMap('name')
==>{name=[A Ambainis]}
```

g.V().valueMap(true)

```
gremlin> g.V(594).valueMap(true)
==>{name=[A Ambainis], id=594, authorId=[199], label=Author}
```

g.V().valueMap(true,'name')

```
gremlin> g.V(594).valueMap(true,'name')
==>{name=[A Ambainis], id=594, label=Author}
```

Exercise 5: Getting Properties

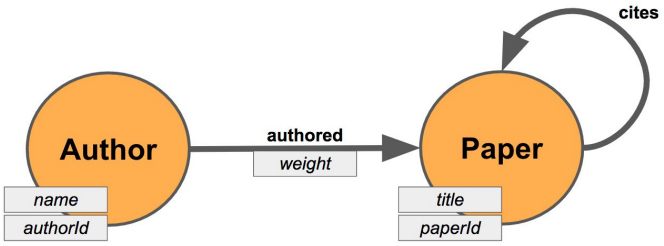
Answer the following questions. Use the Citations graph for writing and testing traversals.

a. What does the Gremlin Console automatically do to the traversal results for you?
b. What is the value of the name property on the Vertex with internal ID: 2142?
c. What is the value of the authorId property on the Vertex with internal ID: 120
d. What is the name of the Author with authorId 46?
e. What is the title of the Paper with internal ID 4110?
f. What is the title of the Paper with paperId 571?
g. Return the map of values, internal ones and properties, for the paper with the title: "Quantum decision theory"

Traversing Outbound Edges

g.V(#).outE()

```
gremlin> g.V(594).outE()
==>e[4912][594-authored->3036]
==>e[5796][594-authored->4242]
==>e[5508][594-authored->3876]
==>e[5931][594-authored->4407]
==>e[5964][594-authored->4434]
==>e[4815][594-authored->2895]
```



g.V(#).outE().valueMap()

```
gremlin> g.V(594).outE().valueMap()
==>{weight=0.4}
==>{weight=0.2857142857}
==>{weight=0.4}
==>{weight=0.6666666667}
==>{weight=0.3333333333}
==>{weight=0.1666666667}
```

g.V(#).outE().valueMap(true)

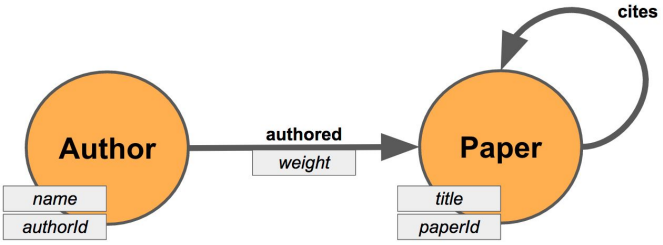
```
gremlin> g.V(594).outE().valueMap(true)
==>{weight=0.4, id=4912, label=authored}
==>{weight=0.2857142857, id=5796, label=authored}
==>{weight=0.4, id=5508, label=authored}
==>{weight=0.6666666667, id=5931, label=authored}
==>{weight=0.3333333333, id=5964, label=authored}
==>{weight=0.1666666667, id=4815, label=authored}
```

g.V(#).outE().valueMap(true)

```
gremlin> g.V(594).outE().valueMap(true)
==>{weight=0.4, id=4912, label=authored}
==>{weight=0.2857142857, id=5796, label=authored}
==>{weight=0.4, id=5508, label=authored}
==>{weight=0.6666666667, id=5931, label=authored}
==>{weight=0.3333333333, id=5964, label=authored}
==>{weight=0.1666666667, id=4815, label=authored}
```

g.V(#).out()

```
gremlin> g.V(594).out()
==>v[3036]
==>v[4242]
==>v[3876]
==>v[4407]
==>v[4434]
==>v[2895]
```



g.V(#).out().valueMap()

```
gremlin> g.V(594).out().valueMap(true)
==>{id=3036, title=[Multiparty ...], label=Paper, paperId=[207]}
==>{id=4242, title=[Worst case ...], label=Paper, paperId=[609]}
==>{id=3876, title=[Nonlocal Qu...], label=Paper, paperId=[487]}
==>{id=4407, title=[Provable Ad...], label=Paper, paperId=[664]}
==>{id=4434, title=[On symmetri...], label=Paper, paperId=[673]}
==>{id=2895, title=[Quantum wal...], label=Paper, paperId=[160]}
```

g.V(#).out().valueMap()

```
gremlin> g.V().hasLabel('Author').
......1>       has('name','A Ambainis').out().valueMap(true)
==>{id=3036, title=[Multiparty ...], label=Paper, paperId=[207]}
==>{id=4242, title=[Worst case ...], label=Paper, paperId=[609]}
```

```
==>{id=3876, title=[Nonlocal Qu...], label=Paper, paperId=[487]}
==>{id=4407, title=[Provable Ad...], label=Paper, paperId=[664]}
==>{id=4434, title=[On symmetri...], label=Paper, paperId=[673]}
==>{id=2895, title=[Quantum wal...], label=Paper, paperId=[160]}
```

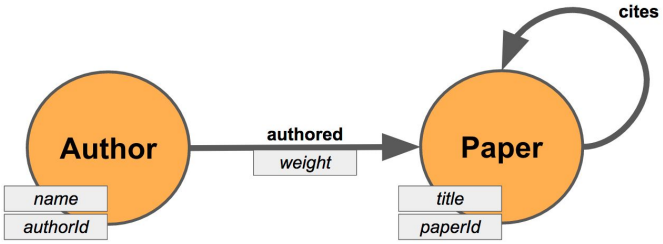
Traversing Inbound Edges

g.V(#).valueMap(true)

```
gremlin> g.V(2424).valueMap(true)
==>{id=2424, title=[A quantum s...], label=Paper, paperId=[3]}
```

g.V(#).inE()

```
gremlin> g.V(2424).inE()
==>e[4476][9-authored->2424]
==>e[4477][12-authored->2424]
==>e[4478][15-authored->2424]
```



g.V(#).inE().valueMap(true)

```
gremlin> g.V(2424).inE().valueMap(true)
==>{weight=0.5, id=4476, label=authored}
==>{weight=0.3333333333, id=4477, label=authored}
==>{weight=0.1666666667, id=4478, label=authored}
```

g.V(#).in().valueMap()

```
gremlin> g.V(3975).in().valueMap()
==>{name=[OG Zabaleta], authorId=[607]}
==>{name=[CM Arizmendi], authorId=[608]}
==>{title=[Quantum decision making by social agents], paperId=[627]}
==>{title=[N-person quantum Russian roulette], paperId=[669]}
==>{title=[Quantum Dating Market], paperId=[656]}
```

g.V(#).in(label).valueMap()

```
gremlin> g.V(3975).in('authored').valueMap()
==>{name=[OG Zabaleta], authorId=[607]}
==>{name=[CM Arizmendi], authorId=[608]}
```

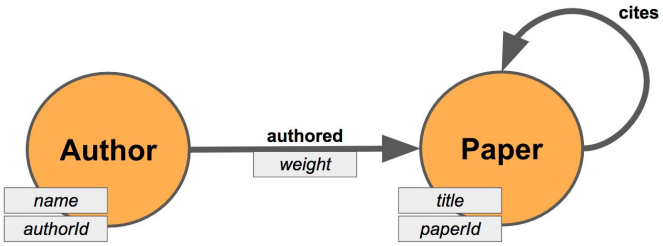
g.V(#).in(label).valueMap()

```
gremlin> g.V(3975).in('cites').valueMap()
==>{title=[Quantum decision making by social agents], paperId=[627]}
==>{title=[N-person quantum Russian roulette], paperId=[669]}
==>{title=[Quantum Dating Market], paperId=[656]}
```

Traversing Inbound & Outbound Edges

g.V(#).bothE()

```
gremlin> g.V(3975).bothE()
==>e[8667][3975-cites->2529]
==>e[8668][3975-cites->2550]
==>e[8669][3975-cites->3714]
==>e[5591][1818-authored->3975]
==>e[5592][1821-authored->3975]
==>e[9330][4296-cites->3975]
==>e[9717][4422-cites->3975]
==>e[9655][4383-cites->3975]
```



g.V(#).both(label)

```
gremlin> g.V(3975).both('cites')
==>v[2529]
==>v[2550]
==>v[3714]
==>v[4296]
==>v[4422]
==>v[4383]
```

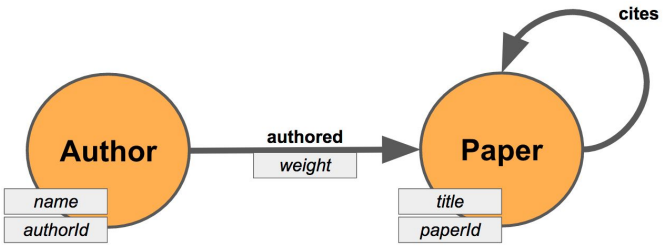
g.V(#).both(label).in(label)

```
gremlin> g.V(3975).both('cites').in('authored')
==>v[153]
==>v[207]
==>v[210]
==>v[213]
==>v[1317]
==>v[1320]
==>v[1320]
==>v[2211]
==>v[1833]
==>v[1821]
==>v[1818]
```

g.V(#).both(label).in(label).valueMap()

```
gremlin> g.V(3975).both('cites').in('authored').valueMap()
==>{name=[DA Meyer], authorId=[52]}
==>{name=[J Eisert], authorId=[70]}
==>{name=[M Wilkens], authorId=[71]}
```

```
==>{name=[M Lewenstein], authorId=[72]}
==>{name=[VI Yukalov], authorId=[440]}
==>{name=[D Sornette], authorId=[441]}
==>{name=[D Sornette], authorId=[441]}
==>{name=[V Yukalov], authorId=[738]}
==>{name=[P Frackiewicz], authorId=[612]}
==>{name=[CM Arizmendi], authorId=[608]}
==>{name=[OG Zabaleta], authorId=[607]}
```



g.V(#).both(label).in(label).dedup().valueMap()

```
gremlin> g.V(3975).both('cites').
......1>           in('authored').dedup().valueMap()
==>{name=[DA Meyer], authorId=[52]}
==>{name=[J Eisert], authorId=[70]}
==>{name=[M Wilkens], authorId=[71]}
==>{name=[M Lewenstein], authorId=[72]}
==>{name=[VI Yukalov], authorId=[440]}
==>{name=[D Sornette], authorId=[441]}
==>{name=[V Yukalov], authorId=[738]}
==>{name=[P Frackiewicz], authorId=[612]}
==>{name=[CM Arizmendi], authorId=[608]}
==>{name=[OG Zabaleta], authorId=[607]}
```

Exercise 6: Traversing Edges

Answer the following questions. Use the Citations graph for writing and testing traversals.

a. What does the Gremlin Console automatically do to the traversal results for you?
b. What is the title for the paper with a paperId value of 119.
c. Get the authors for the paper with paperId value of 119. How many are there?
d. How many other papers did those authors collaborate on?
e. For the Paper with paperId 91 what Papers does it cite? What Papers cite it?
f. For the Paper with the title “A survey of quantum games”:
    i. What is its internal ID?
    ii. What is its paperId value?
    iii. How many Papers either cite or are cited by it?

Gremlin's Predicates

Predicates take an object and return a true or false.

Predicate	Description
eq(object)	Is the incoming object equal to the provided object?
neq(object)	Is the incoming object not equal to the provided object?
lt(number)	Is the incoming number less than the provided number?
lte(number)	Is the incoming number less than or equal to the provided number?
gt(number)	Is the incoming number greater than the provided number?
gte(number)	Is the incoming number greater than or equal to the provided number?
inside(number,number)	Is the incoming number greater than the first provided number and less than the second?
outside(number,number)	Is the incoming number less than the first provided number or greater than the second?
between(number,number)	Is the incoming number greater than or equal to the first provided number and less than the second?
within(objects...)	Is the incoming object in the array of provided objects?
without(objects...)	Is the incoming object not in the array of the provided objects?
not(predicate())	Returns the logical negation of the predicate.
and(predicate(),...)	Predicates can be *and*'d together.
or(predicate(),...)	Predicates can be *or*'d together.

Filtering

g.V().has(key,predicate).valueMap()

```
gremlin> g.V().has('paperId',lte(5)).valueMap()
==>{title=[Utility of information game theoretic approach], paperId=[1]}
==>{title=[A GLOBAL THEORY FOR LINEAR QUADRATIC DIFFER...], paperId=[2]}
==>{title=[A quantum shuffling game for teaching stati...], paperId=[3]}
==>{title=[A max-min differential game in Hilbert space], paperId=[4]}
==>{title=[Statistical decision theory for quantum systems], paperId=[5]}
```

g.V().hasLabel(label).values(key).is(predicate)

```
gremlin> g.V().hasLabel('Author').values('authorId').is(gt(800))
==>801
==>802
==>803
==>804
==>805
==>806
```

For the authors of paperId 119, what other papers have they written, not including paperId 119?

```
gremlin> g.V().has('paperId',119).as('a').
......1>       in('authored').out('authored').as('b').
......2>       where('b',neq('a')).dedup().values('title')
==>Quantum game in matrix strategies
==>Quantum Game with Restricted Matrix Strategies
```

Exercise 7: Filtering

Answer the following questions. Use the Citations graph for writing and testing traversals.

a. What does the Gremlin Console automatically do to the traversal results for you?
b. List the titles of the papers with paperId values between 10 and 20
c. List the names of the authors that start with A.
d. What authors wrote papers with J Kempe co-authors.

Introduction to Gremlin

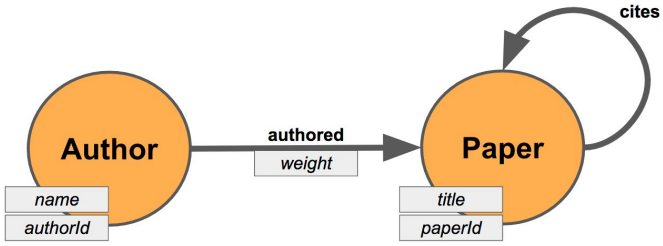
Online Course - July 21, 2017

Section 3: Mutating the Graph

All your C.U.D. are belong to us

Topics to cover:

- Graph vs Traversal Review
- Adding Data
- Editing Data
- Removing Data



Adding Vertices

Graph: *graph.addVertex(Type,label,key,value…)*

```
gremlin> bfrank = graph.addVertex(label,'Author',
......1>                          'authorId',1001,
......2>                          'name','B Frank')
==>v[1]
```

Traversal: *g.addV(label).property(key,value)*

```
gremlin> g.addV('Author').property('authorId',1002).
......1>                  property('name','C Delay')
==>v[5]
```

"Safe 1": *g.V().has(label,key,value).tryNext().orElseGet({g.addV(T.label,label,key,value}.next())*

```
gremlin> g.V().has('paperId',2002).tryNext().
......1>                        orElseGet({
......2>                                  g.addV('Paper').
......3>                                    property('paperId',2002).
......4>                                    property('title','Wee Delay').
......5>                                    next()
......6>                                  })
==>v[10]
```

"Safe 2": *g.V().has(label,key,value).fold().coalesce(unfold(),addV(T.label,label,key,value))*

```
gremlin> g.V().has('paperId',2003).fold().
......1>                        coalesce(
......2>                                 unfold()
......3>                               , addV('Paper').
......4>                                   property('paperId',2003).
......5>                                   property('title','Outa Time'))
==>v[12]
```

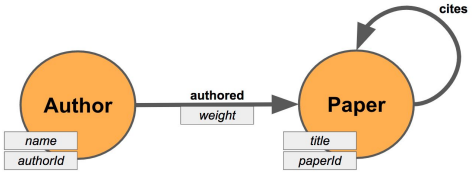
Add another Paper for addEdge() example

```
gremlin> p2001 = graph.addVertex(label,'Paper',
......1>                         'paperId',2001,
......2>                         'title','Let Me Be Frank')
==>v[14]
```

Adding Edges

Graph: *vertex1.addEdge(label, vertex2, key, value, ...)*

```
gremlin> bfrank.addEdge('authored',p2001,'weight',1)
==>e[19][1-authored->14]
```



Traversal: *g.V().has(key,value).as(alias1).*
V().has(key,value).as(alias2).
addE(label).from(alias1).to(alias2).property(key,value)

```
gremlin> g.V().has('authorId',1002).as('a').
......1>   V().has('paperId',2002).as('p').
......2>       addE('authored').from('a').to('p').
......3>       property('weight',1)
==>e[20][5-authored->10]
```

Graph: *vertex.property(key, value)*

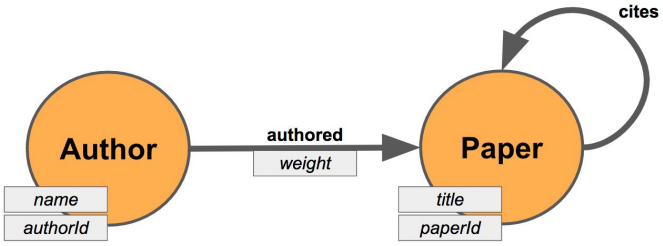
```
gremlin> g.V().has('authorId',1002).as('a').
......1>   V().has('paperId',2002).as('p').
......2>       addE('authored').from('a').to('p').
......3>       property('weight',1)
==>e[20][5-authored->10]
```

Exercise 8: Adding Data

Answer the following questions. Use the Citations graph for writing and testing traversals.

a. What is the difference between a Graph and a TraversalSource?
b. When mutating a graph, should you use the Graph or the TraversalSource?
c. Add an Author with name: **M A Rodriguez**, authorId: **900**
d. Add an Author with name: **J H Watkins**, authorId: **901**
e. Add a Paper with title: **Quantum Walks with Gremlin**, paperId: **700**
f. For both of the Authors that were added, add an edge with the label: **authored** connecting them to the Paper that was added. For both edges set the **weight** property to 0.5.

Adding and Editing Properties



Graph: *vertex.property(key, value)*

```
gremlin> bfrank.property('age',27)
==>vp[age->27]
```

Verify with Traversal

```
gremlin> bfrank
==>v[1]
gremlin> g.V(bfrank).valueMap()
==>{name=[B Frank], authorId=[1001], age=[27]}
```

Traversal: *g.V().has(key,value).property(key,value)*

```
gremlin> g.V().has('authorId',1002).property('age',35)
==>v[5]
```

Traversal: *g.V().has(key,value).property(list,key,value)*

```
gremlin> g.V().has('authorId',1002).property(list,'name','Charles')
==>v[5]
```

Verify with a traversal

```
gremlin> g.V().has('authorId',1002).values('name')
==>C Delay
==>Charles
```

Traversal: *g.V().has(key,value).property(set,key,value)*

```
gremlin> g.V().has('authorId',1002).
......1>          property(set,'email','c@gmail.com').
......2>          property(set,'email','c@hotmail.com')
==>v[5]
```

Traversal: *g.V().has(key,value).property(single,key,value)*

```
gremlin> g.V().has('authorId',1002).
......1>         property(single,'name','C Delay')
==>v[5]
```

Change a property with Traversal: *g.V().has(key,value).property(key,value)*

```
gremlin> g.V().has('authorId',1002).property('age',29)
==>v[5]
```

Reset a Set property with Traversal: *g.V().has(key,value).property(set,key,value)*

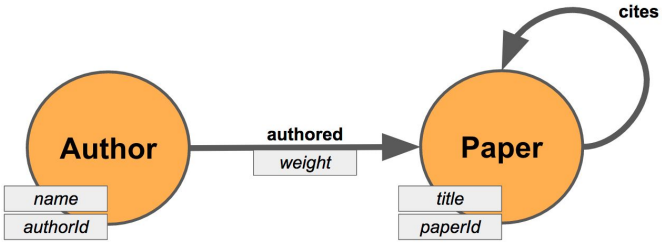
```
gremlin> g.V().has('authorId',1002).
......1>       property('email','c@yahoo.com')
==>v[5]

gremlin> g.V().has('authorId',1002).valueMap()
==>{name=[C Delay], authorId=[1002], age=[29], email=[c@yahoo.com]}
```

Removing Properties

Graph: *vertex.property(key).remove()*

```
gremlin> bfrank.property('age').remove()
==>null
gremlin> g.V(bfrank).valueMap()
==>{name=[B Frank], authorId=[1001]}
```



Traversal: *g.V().has(key,value).properties(key).drop()*

```
gremlin> g.V().has('authorId',1002).properties('age').drop()
gremlin> g.V().has('authorId',1002).valueMap()
==>{name=[B Frank], authorId=[1001]}
```

Dropping Edges

Traversal: *g.V().has(key,value).out(label).drop()*

```
gremlin> g.V().has('authorId',1002).outE()
==>e[20][5-authored->10]
gremlin> g.V().has('authorId',1002).outE().drop()
gremlin> g.V().has('authorId',1002).outE()
gremlin>
```

Removing Vertices

Graph: *vertex.remove()*

```
gremlin> g.V(bfrank).valueMap()
==>{name=[B Frank], authorId=[1001]}
gremlin> bfrank.remove()
==>null
gremlin> g.V().has('authorId',1001)
gremlin>
```

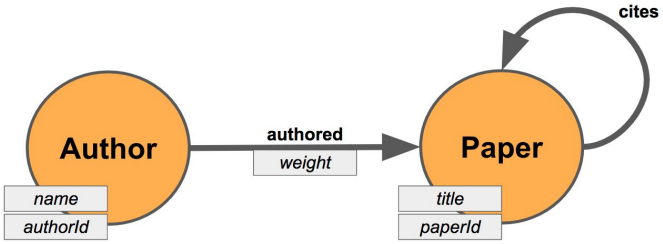
Traversal: *g.V().has(key,value).drop()*

```
gremlin> g.V().has('authorId',1002).valueMap()
==>{name=[C Delay], authorId=[1002], email=[c@yahoo.com]}
gremlin> g.V().has('authorId',1002).drop()
gremlin> g.V().has('authorId',1002).valueMap()
gremlin>
```

Removing a Graph

Traversal: *g.E().drop(); g.V().drop()*

```
gremlin> gclassic.E().count()
==>6
gremlin> gclassic.E().drop()
gremlin> gclassic.E().count()
==>0
gremlin> gclassic.V().count()
==>6
gremlin> gclassic.V().drop()
gremlin> gclassic.V().count()
==>0
```



Recover the Classic graph Elements by restarting the Docker container.

Exercise 9: Editing / Removing Data

Answer the following questions. Use the Citations graph for writing and testing traversals.

a. Remove the name property from one of the two Authors which you added in Exercise 8.
b. Remove the Paper vertex that doesn't have a name without using a Traversal.
c. For the remaining new Author, change the weight on the authored edge to 1.
d. For the remaining new Author, change their name to N A Smith.
e. Remove the two Author vertices where were created in Exercise 8
f. Drop The Crew. (If you want The Crew back, just exit the Gremlin Console to stop the Container and restart).

Introduction to Gremlin

Online Course - July 21, 2017

Section 4: Common Transformations

Topics to cover:

- Projections
- Grouping
- Ordering
- Aggregations
- (a little theory)

Projections

Assign a variable name to a vertex with: *vertex = g.V().has(key,value).next(); null*

```
gremlin> zhi = g.V().has('name','H Zhi').next(); null
==>null
```

List of the *name* property values

```
gremlin> g.V(zhi).values('name')
==>H Zhi
```

Map of the *name* properties and values including the vertex label and ID

```
gremlin> g.V(zhi).valueMap(true,'name')
==>{name=[H Zhi], authorId=[684], id=2049, label=Author}
```

g.V().....as('alias1')...as('alias2')....select('alias1','alias2').by().by()

```
gremlin> g.V().has('paperId',104).
......1>               in('authored').as('a').
......2>               out('authored').as('t').
......3>               select('a','t').by('name').by('title')
==>{a=YJ Han, t=W state and Greenberger-Horne-Zeilinger state...}
==>{a=YS Zhang, t=Qunatum Parrondo's games constructed by qua...}
==>{a=YS Zhang, t=W state and Greenberger-Horne-Zeilinger sta...}
==>{a=YS Zhang, t=Optical realization of quantum gambling mac...}
==>{a=YS Zhang, t=Quantum strategies of quantum measurements}
==>{a=YS Zhang, t=Quantum Game Theory}
==>{a=GC Guo, t=Quantum Parrondo game based on a quantum ratc...}
==>{a=GC Guo, t=Qunatum Parrondo's games constructed by quant...}
==>{a=GC Guo, t=W state and Greenberger-Horne-Zeilinger stat...}
==>{a=GC Guo, t=Optical realization of quantum gambling mach...}
==>{a=GC Guo, t=Quantum strategies of quantum measurements}
```

g.V().....project('alias1','alias2').by().by()

```
gremlin> g.V().hasLabel('Paper').limit(5).
......1>          project('title','authorCount','citesCount').
......2>               by(values('title')).
......2>               by(inE('authored').count()).
......2>               by(outE('cites').count())
==>{title=Outa Time, authorCount=0, citesCount=0}
==>{title=Bell violations through independe..., authorCount=1, citesCount=1}
==>{title=Photon shell game in three-reson..., authorCount=15, citesCount=0}
==>{title=Non-adaptive measurement-based qu..., authorCount=4, citesCount=2}
==>{title=QUANTUM MECHANISM HELPS AGENTS CO..., authorCount=1, citesCount=7}
```

Grouping

Group all of the vertices in the graph by their label and return their count.

```
gremlin> g.V().group().by(label).by(count())
==>{Author=806, Paper=685}
```

Group all of the vertices in the graph by their label and return their count.

```
gremlin> g.V().groupCount().by(label)
==>{Author=806, Paper=685}
```

Ordering

For the author N Gisin, list the paper titles in alphabetical order.

```
gremlin> gisin = g.V().has('authorId',39).next(); null
==>null
gremlin> g.V(gisin).out('authored').
......1>            order().by('title').
......2>            values('title')
==>Guess Your Neighbor’s Input: A Multipartite Nonlocal Game with...
==>Optimal eavesdropping in quantum cryptography. I. Information ...
==>Quantum cryptography with coherent states
==>Quantum solution to the Byzantine agreement problem
```

For the author N Gisin, list the paper titles in reverse alphabetical order.

```
gremlin> g.V(gisin).out('authored').
......1>             order().by('title',decr).
......2>             values('title')
==>Quantum solution to the Byzantine agreement problem
==>Quantum cryptography with coherent states
==>Optimal eavesdropping in quantum cryptography. I. Information ...
```

```
==>Guess Your Neighbor's Input: A Multipartite Nonlocal Game with...
```

Aggregations

How many vertices are in the Citations graph?

```
gremlin> g.V().count()
==>1491
```

How many papers has authorId 114 authored?

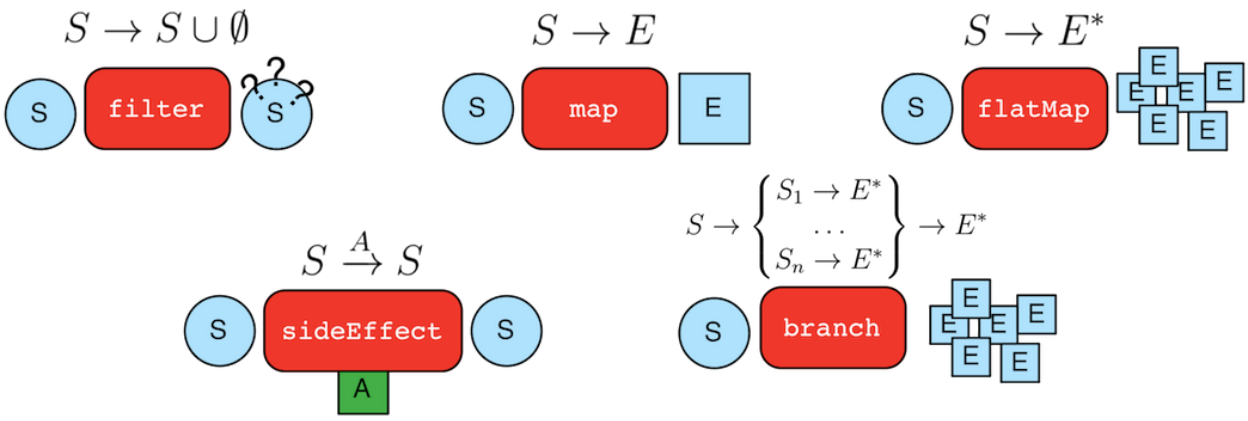
```
gremlin> g.V().has('authorId',114).out('authored').count()
==>48
```

What is the maximum value of the authorId property?

```
gremlin> g.V().values('authorId').max()
==>806
```

A Wee Bit o' Theory

Gremlin Step Superclasses



Exercise 10: Transformations

Answer the following questions. Use the Citations graph for writing and testing traversals.

a. For paper 101, list the name of the authors and the titles of all of their papers.
b. What are the labels in the Grateful Dead graph (traversal: ggrateful) and how many vertexes are there for each label?
c. List the titles for the papers by X Xu in alphabetical order.
d. Find the minimum and maximum paperId values.