All the "Vs" of big (graph) data

Dr. Ora Lassila

Principal Technologist, Amazon Neptune

Co-chair, W3C RDF-star WG

former Chief Scientist, Nokia Venture Partners

former W3C Fellow

Semantic Arts' 7th Annual **Data-Centric Architecture Forum**, June 2025

aws

Who am I, and what have I done?

Current:

Principal Technologist, Amazon Neptune (AWS)
Co-chair, W3C RDF-star Working Group

Past:

State Street, Pegasystems, Nokia,
MIT, CMU, Helsinki University of Technology, …

Education:

Ph.D (D.Sc) CS & AI, Helsinki University of Technology

Semantic Web vision

RDF

KR for NASA Deep Space 1

2 daughters

43k+ citations

Grand Prize of Usenix
Obfuscated C Code Contest

aws

This talk is about scalability

But the real challenge of scalability is not where you think it is

Setting the stage

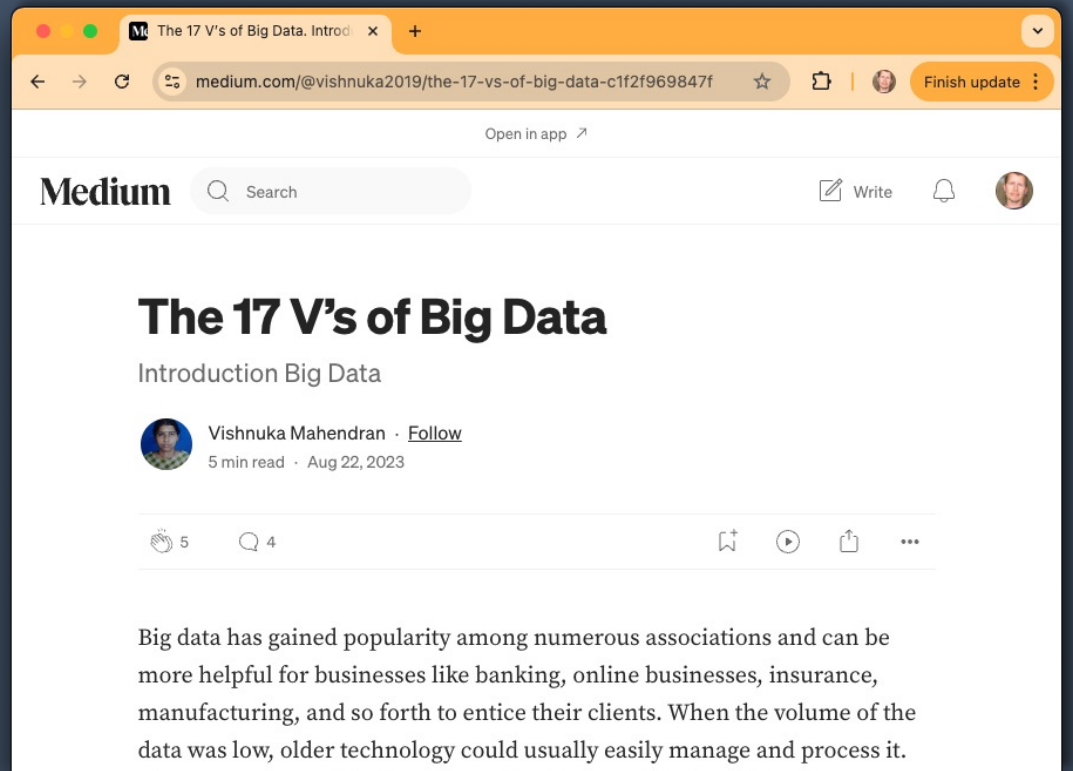
"Big data"

The term was coined in the 1990s

Originally "3 Vs" (volume, velocity, variety)

- later, many more "Vs"…

Seriously…?



Bottom line is that much of the good that we knew about data & databases was sacrificed for the "Vs"

What happened?

Rise of “NoSQL” databases

Features like schema, referential integrity, joins, etc., all moved from the database to the application code (or were more or less ditched)

1. implemented *ad hoc*
2. schema implicit in procedural code
   → no real opportunity to manage complex models

Result: more velocity (with caveats), more volume (maybe), but ***less variety*** (or at least no good way to handle variety)

REALLY BAD IDEA:

Features like schema, referential integrity, joins, etc., all moved from the database to the application code (or were more or less ditched)

1. implemented *ad hoc*
2. schema implicit in procedural code

   → no real opportunity to manage complex models

Result: more velocity (with caveats), more volume (maybe), but ***less variety*** (or minimally, no good way to handle variety)

What about relational?

The world does not always fit in rigid tables

or

You end up with millions of columns

What about relational?

The world does not ~~always~~ fit in rigid tables

~~or~~ *and*

You end up with millions of columns

Hard to understand, hard to maintain, hard to evolve

- also: complex queries (too many joins, etc.)

aws

Enter graphs and ontologies

Some background: Graphs, ontologies, RDF, etc.

Graphs have been around since the 1730s

Ontologies have been around since the very early 1900s

RDF emerged in the late 1990s, but should be seen as coming from the frame-based KR tradition (since the 1970s)

Long despised by database folks

- the relational algebra is not particularly useful for optimizing this stuff
- also: the database people and KR people did not talk to one another…

Introduction to RDF

Simple KR language, cornerstone of the "Semantic Web stack"

Widely adopted and deployed

- about half of all Web pages contain some embedded RDF
- all Adobe documents contain some RDF
- etc.

"RDF has turned out to be the most catastrophically successful failure there is." – Charles Ivie (AWS)

Does RDF scale? (The first two "Vs")

(RDF decomposes into "triples", effectively these are the graph edges)

Early Semantic Web conferences had a "billion triple challenge"

Several years ago, AllegroGraph announced they had ingested 1T triples

Single Amazon Neptune cluster now scales up to about 0.5T triples

Here is one example of how RDF scales…

Inventory knowledge graph of Amazon Fulfillment is used to

- investigate fulfillment processes
- investigate lost & found -issues
- improve precision of product recalls

Extends the PROV-O ontology: models the end-to-end logistics process as a form of provenance

Runs on multiple (federated) Neptune clusters

Size:

1T+ triples

4B new triples per day

Queries (p95):

< 50 ms to find a node

< 1 s to retrieve the whole path

Here is one example of how RDF scales…

I think we have the first two “Vs” covered!

Size:

- 1T+ triples
- 4B new triples per day

Queries (p95):

- < 50 ms to find a node
- < 1 s to retrieve the whole path

But what about the other “Vs”…?

But what about the other “Vs”…?

Variety

Veracity

Value

Viability, Variability, Viscosity, Validity, Vulnerability, Volatility, blah blah blah…

Variety RDF and ontologies are well suited to data integration

Veracity Complex models and queries supported

Value Ontologies make complex data easier to understand

Interoperability tends to increase variety

…and variety implies **complexity**

Variety

Veracity Easy to capture lineage and provenance in RDF

Value Logical inference supports explainable AI

Declarative, accessible ontologies promote openness

Variety

Veracity

Value

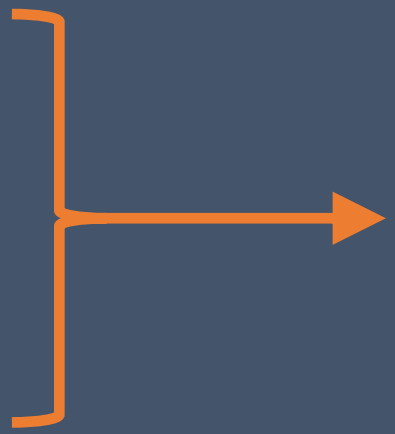
I'd rather talk about cost here… (the cost of not using the Semantic Web stack, that is)

- cost of maintenance
- cost of technical debt
- cost of redundancy
- etc.

Variety

Veracity

Value



Semantics as the enabler

These do not start with a “V”,
but should be included:

Complexity

Interoperability

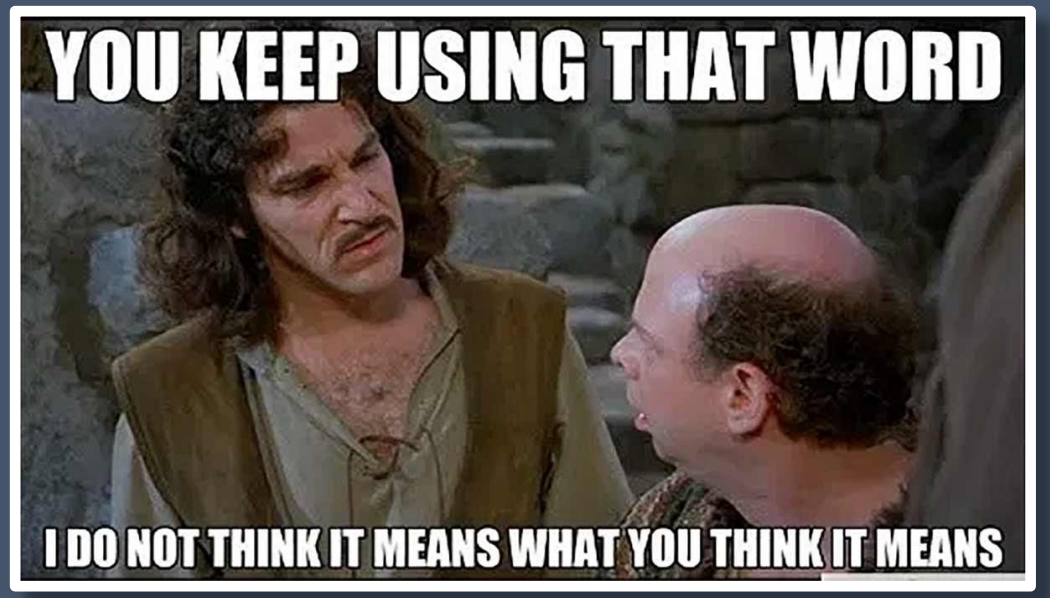
Semantics, and what is that anyway?

Semantics

Overused word, thrown around liberally

Most people do not really know what the term means

- (cue scene from “The Princess Bride”)



1. Separating formal semantics from one’s own (human) interpretation?
2. Where does semantics come from?

aws

Separating formal semantics from one's own interpretation

Example: JSON

- people say "my data is just JSON" and "JSON is easy to understand"
- but JSON has no semantics (at all), so any "understanding" is based on some external semantics (typically not declarative or accessible)

“JSON is easy to understand”

```
{
        “first name”: “Ora”,
        “family name”: “Lassila”,
        “degree”: “Ph.D”,
        “place of birth”: “Helsinki”,
        “hobbies”: [ “photography”, “scale models” ]
}
```

Now you are going to tell me you “understand” this data?

“JSON is easy to understand”

```
{
        “etunimi”: “Ora”,
        “sukunimi”: “Lassila”,
        “tutkinto”: “TKT”,
        “syntymäpaikka”: “Helsinki”,
        “harrastukset”: [ “valokuvaus”, “pienoismallit” ]
}
```

Ostensibly this JSON fragment has the same “meaning” as the previous one.

Do you still understand it? And will a machine understand it?

aws

Understandability of JSON is a fallacy

Where does semantics come from?

1. Relationship of data to definitions (ontologies)
2. Relationship of data to some other data
3. Software that interprets data ← "grounding"

We want more of #1 and #2, and (much) less of #3

I see this as a requirement to truly get to "data-centricity"

aws

RDF vs. “the other kinds of graphs”

RDF vs. Labeled Property Graphs

Totally different origins:

- RDF: knowledge representation, Web, open world
- LPG: databases, software development, closed world

RDF is a representation language whereas LPGs are data structures

- if you use RDF as a mere data structure, you are "doing it wrong"
- (and mind you, RDF is only incidentally a graph)

LPGs have good uses, but knowledge graphs should not be one of them

RDF vs. Labeled Property Graphs

A graph is a graph is a graph? *

True, but not really relevant

- similarly, you could say any Turing-complete programming language is just like any other Turing-complete programming language
- you could build your KG system using LPG, but why?

** (with apologies to Gertrude Stein)*

Features of RDF you end up reinventing if you use LPGs

Strong, global identifiers

Predictable (and easy) graph merging

Standardized interchange formats

Schema language (for defining ontologies)

Reasoning

Federated queries

Features of RDF you end up reinventing if you use LPGs

Strong, global identifiers

Predictable (and easy) graph merging

Standardized interchange formats

interoperability

Schema language (for defining ontologies)

Reasoning

Federated queries

complexity

Ora's Rule of Knowledge Graph Implementation

"Any sufficiently sophisticated knowledge graph system built using an LPG contains an ad hoc, informally-specified, bug-ridden implementation of half of RDF." *

** (with apologies to Philip Greenspun re: "Greenspun's Tenth Rule" – look it up)*

aws

Could there be better alignment between RDF and LPGs?

"Project OneGraph" (AWS)

- practical goals: common storage & query language interoperability
- we already have openCypher over RDF → mixed use of RDF and LPG data
- a proposal for LPG-style composite datatypes for RDF (via datatypes)

RDF 1.2 (formerly "RDF-star")

- easier use of reification finally gives us "edge properties"
- RDF 1.2 is more expressive than LPGs (e.g., edges between edges)
- particularly well suited for use cases with cross-cutting aspects

Finale (pronounced “finally” ;-)

Did we miss something? What else needs to scale?

We have speed and size (velocity and volume) covered

If we really apply the Semantic Web stack we can conquer variety

- (including challenges of complexity and interoperability)
- this should also take care of the “things not strings” issue

So what else?

USER EXPERIENCE!

User interfaces do not scale (vis-à-vis expressivity)

We have extremely expressive data, but UI expressivity lags behind

- tabular views dominate
- graph visualization seldom scales, and may not be the answer either
- (also, I am not convinced that LLMs are the answer)

For years I (erroneously) convinced myself that this is not a problem specifically the Semantic Web community should fix

- I am not all that convinced anymore…

My “call to arms” for you:

Velocity and volume already covered

We have solutions for variety, complexity and interoperability

Please make all this technology easier to use!

- I see advances that make developers’ lives easier
- I do not see the same for end users (instead, we are told that AI solves all)

Summary

Big Data & demise of well-established data techniques was a **Bad Idea**

3Vs:

- generally, volume and velocity are not a problem anymore
- with increased complexity, the relational approach falls short, but ontologies and graphs hold a lot of promise → variety covered

I see user interaction and user interfaces as the “final frontier”

Thank you! Any questions?

Contact: ora@amazon.com

Many thanks to:

My colleagues in the Neptune team

Tanya Shigaeva, Juan Sequeda, Lauren Lassila, Mara Owens

Adrian Gschwend and all the members of the W3C RDF-star WG