Fakultas
Ekonomi & Bisnis
Telkom University

Program Studi: Manajemen Bisnis Telekomunikasi & Informatika
Mata Kuliah: Big Data And Data Analytics
Oleh: Tim Dosen

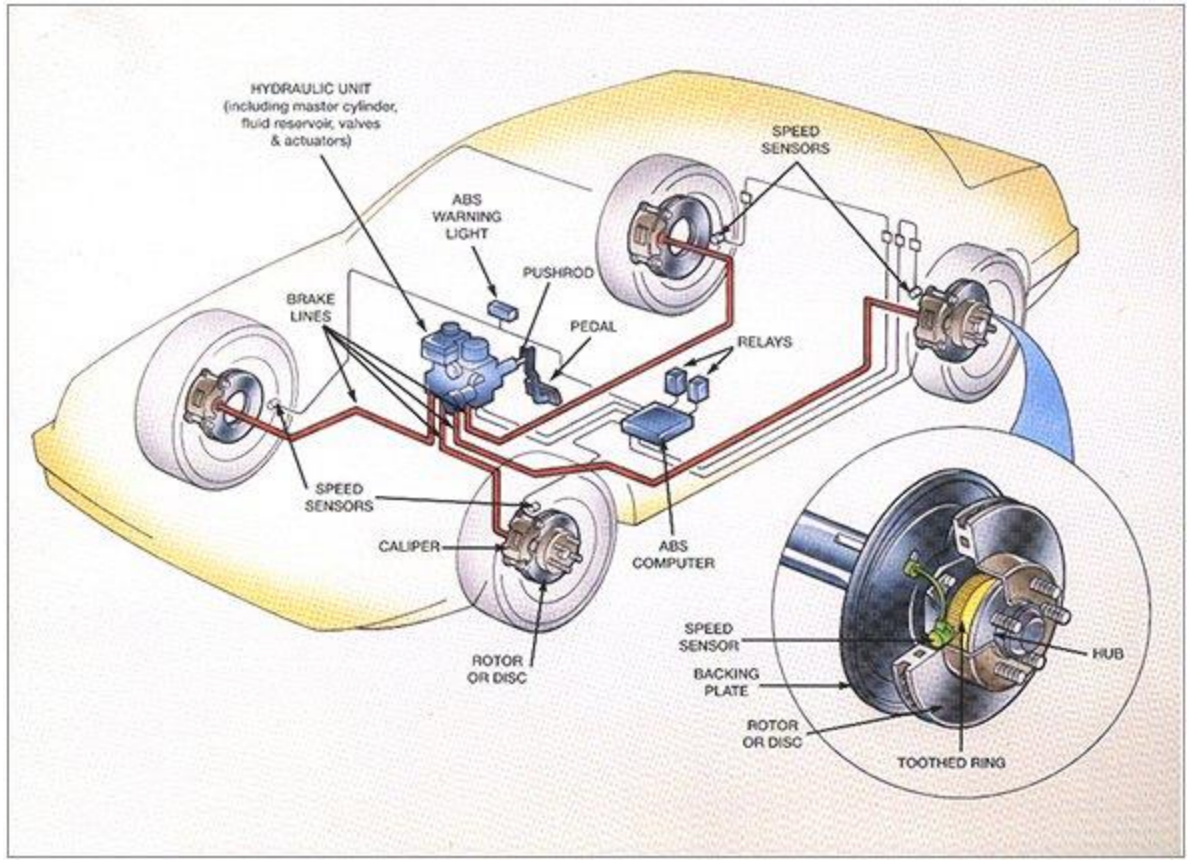
Algorithm, Complexity Theory, and Data Analytics Strategy

Story

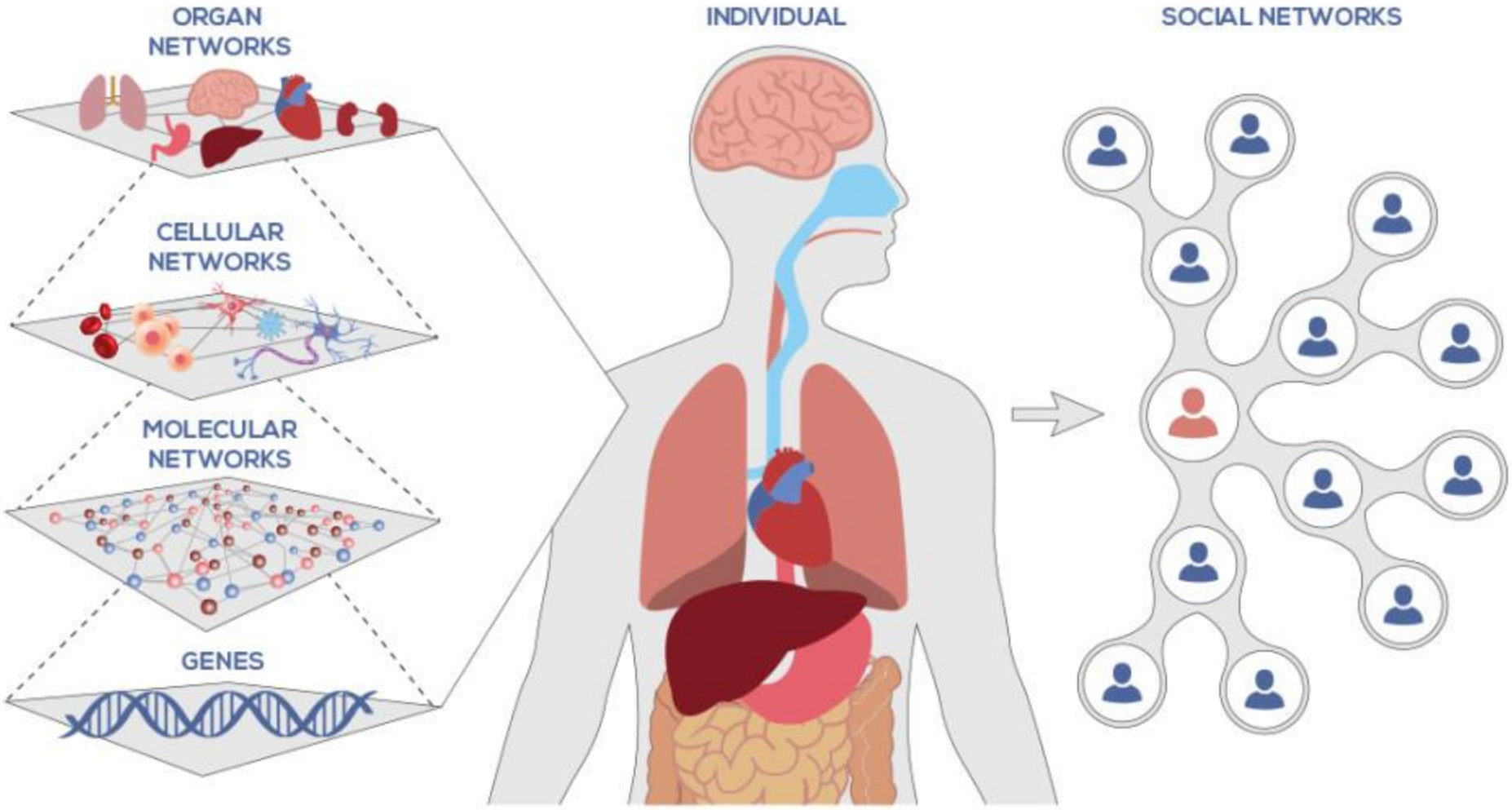
- **"Complexity Science is a double-edged sword in the best possible sense. It is truly "big science" in that it embodies some of the hardest, most fundamental and most challenging open problems in academia. Yet it also manages to encapsulate the major practical issues which face us every day from our personal lives and health, through to global security. Making a pizza is complicated, but not complex. The same holds for filling out your tax return, or mending a bicycle puncture. Just follow the instructions step by step, and you will eventually be able to go from start to finish without too much trouble. But imagine trying to do all three at the same time. Worse still, suppose that the sequence of steps that you follow in one task actually depends on how things are progressing with the other two. Difficult? Well, you now have an indication of what Complexity is all about. With that in mind, now substitute those three interconnected tasks for a situation in which three interconnected people each try to follow their own instincts and strategies while reacting to the actions of the others. This then gives an idea of just how Complexity might arise all around us in our daily lives. "**
- **(Neil Johnson, Simply Complexity p.12)**

Complexity in our daily live

COMPlex?


HYDRAULIC UNIT
(including master cylinder, fluid reservoir, valves & actuators)
SPEED SENSORS
ABS WARNING LIGHT
PUSHROD
BRAKE LINES
PEDAL
RELAYS
SPEED SENSORS
CALIPER
ABS COMPUTER
ROTOR OR DISC
SPEED SENSOR
HUB
BACKING PLATE
ROTOR OR DISC
TOOTHED RING

How about this?

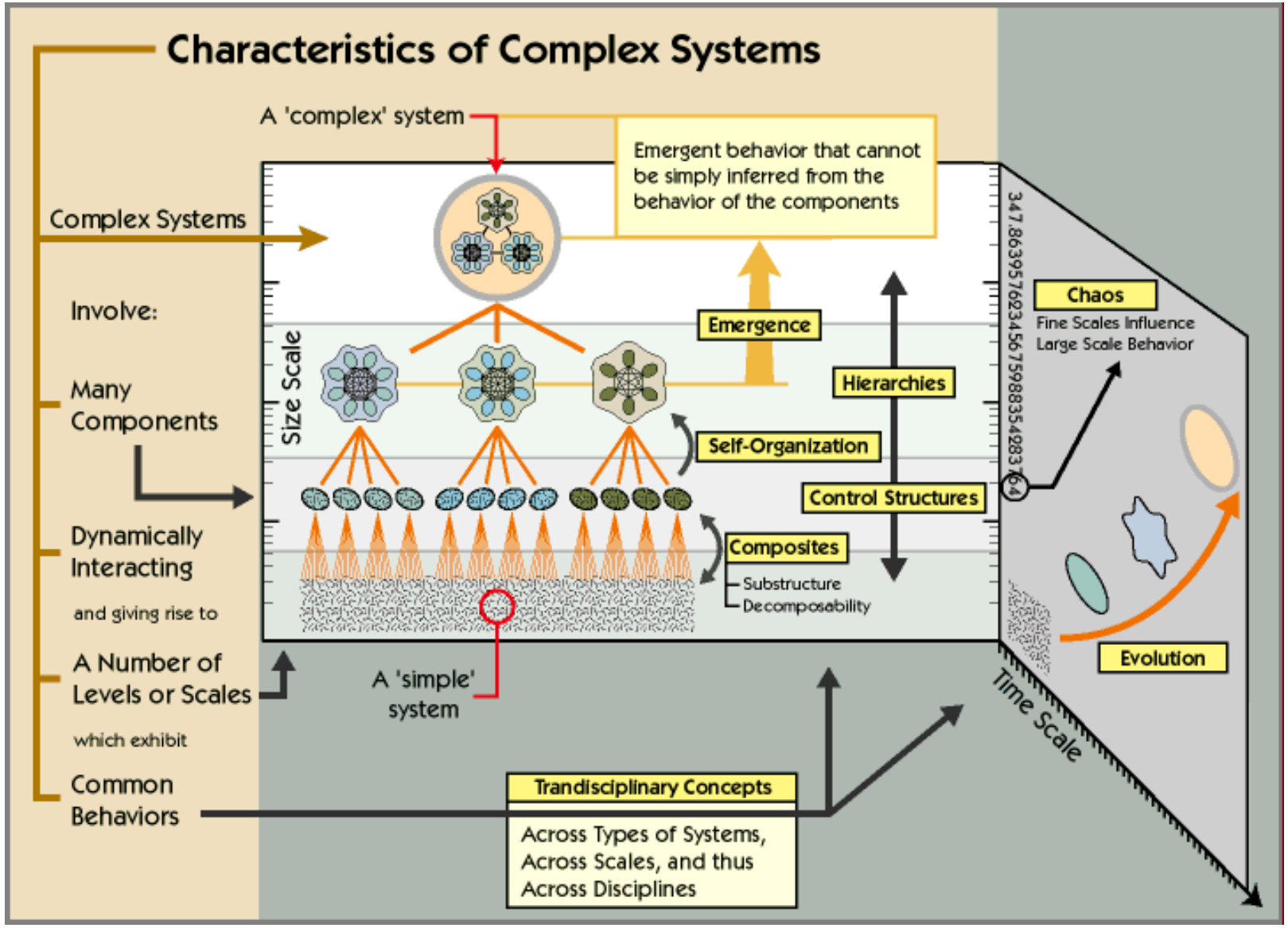


Complexity Theory

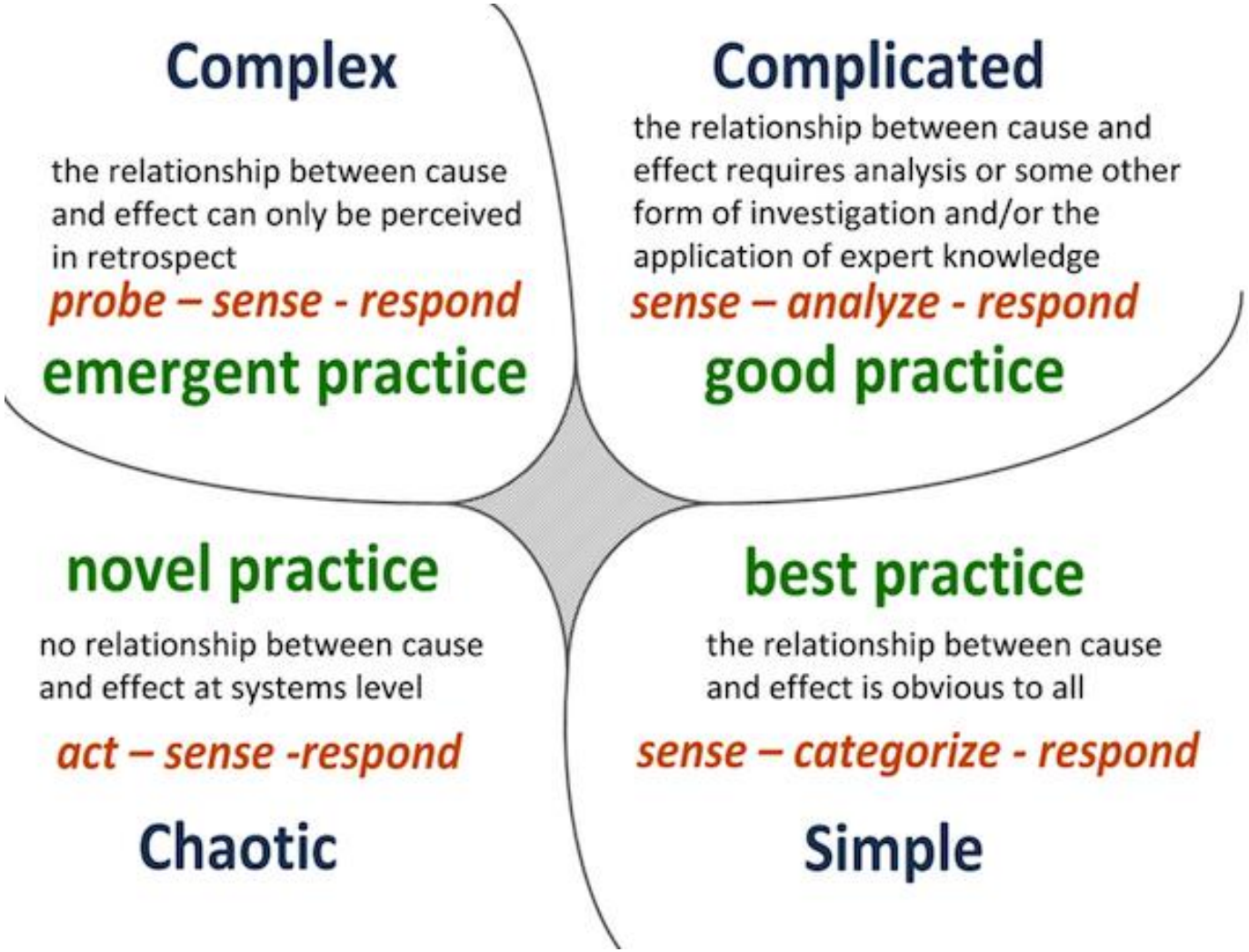
Two Important Dimensions

1. **Space / Size**
2. **Time**

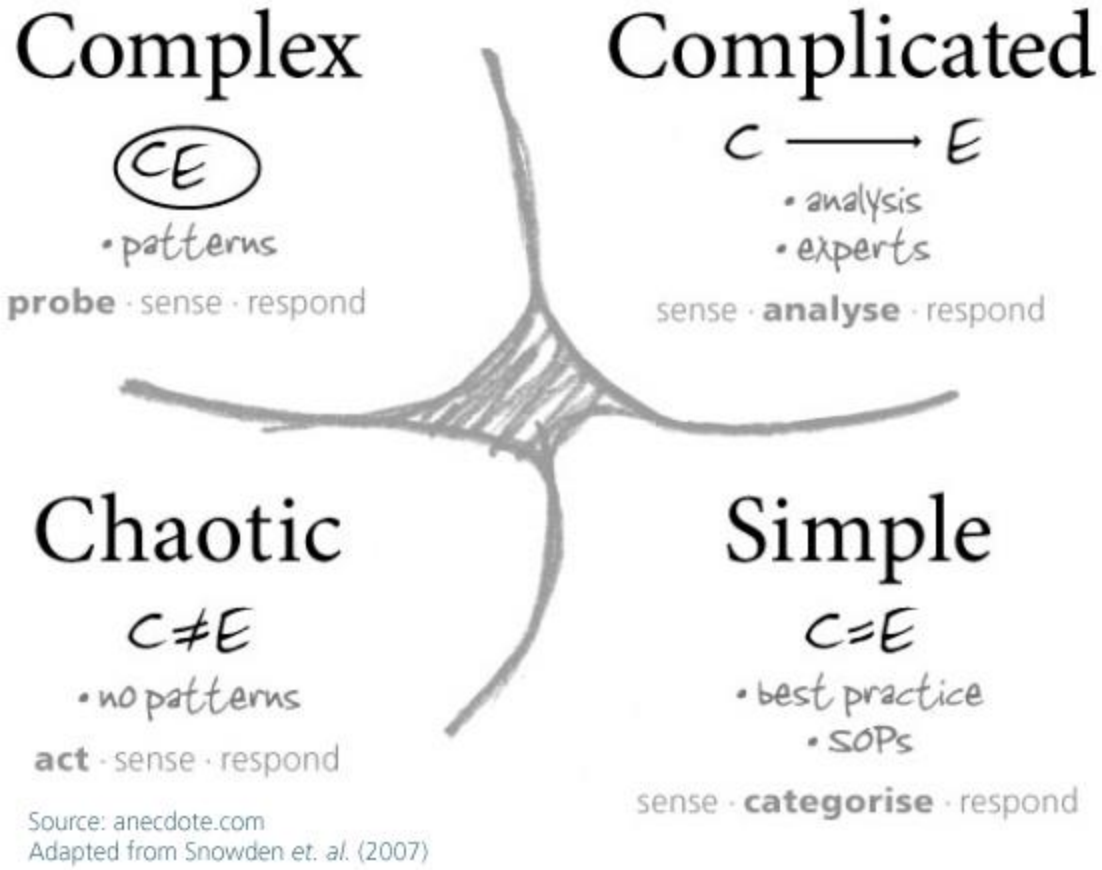
View


Characteristics of Complex Systems
A 'complex' system
Emergent behavior that cannot be simply inferred from the behavior of the components
Complex Systems
Involve:
Many Components
Dynamically Interacting
and giving rise to
A Number of Levels or Scales
which exhibit
Common Behaviors
Size Scale
Emergence
Hierarchies
Self-Organization
Control Structures
Composites
Substructure
Decomposability
A 'simple' system
Chaos
Fine Scales Influence Large Scale Behavior
Evolution
Time Scale
Trandisciplinary Concepts
Across Types of Systems, Across Scales, and thus Across Disciplines

Cynefin Framework (Kih-neh-vihn)



Also CYNEfin framework



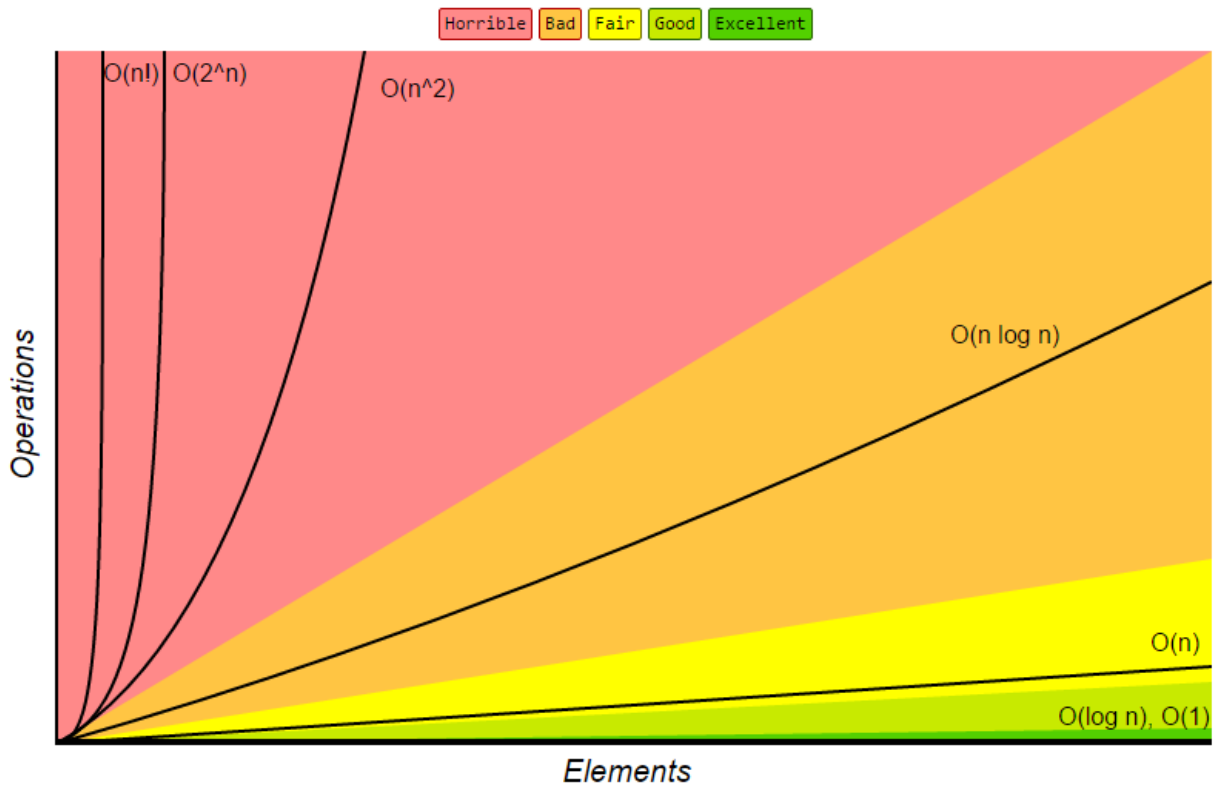
Cynefin framework

- **The framework provides a typology of contexts that guides what sort of explanations or solutions might apply. It draws on research into complex adaptive systems theory, cognitive science, anthropology, and narrative patterns, as well as evolutionary psychology, to describe problems, situations, and systems. It "explores the relationship between man, experience, and context“ and proposes new approaches to communication, decision-making, policy-making, and knowledge management in complex social environments.**

Explanation

- **The Cynefin framework has five domains. The first four domains are:**
- **Obvious - replacing the previously used terminology Simple from early 2014 - in which the relationship between cause and effect is obvious to all, the approach is to *Sense - Categorize - Respond* and we can apply *best* practice.**
- **Complicated, in which the relationship between cause and effect requires analysis or some other form of investigation and/or the application of expert knowledge, the approach is to *Sense - Analyze - Respond* and we can apply *good* practice.**
- **Complex, in which the relationship between cause and effect can only be perceived in retrospect, but not in advance, the approach is to *Probe - Sense - Respond* and we can sense *emergent* practice.**
- **Chaotic, in which there is no relationship between cause and effect at systems level, the approach is to *Act - Sense - Respond* and we can discover *novel* practice.**
- **The fifth domain is Disorder, which is the state of not knowing what type of causality exists, in which state people will revert to their own comfort zone in making a decision. In full use, the Cynefin framework has sub-domains, and the boundary between obvious and chaotic is seen as a catastrophic one: complacency leads to failure.**

Complexity in computing



Data Structure Complexity

bigocheatsheet.com

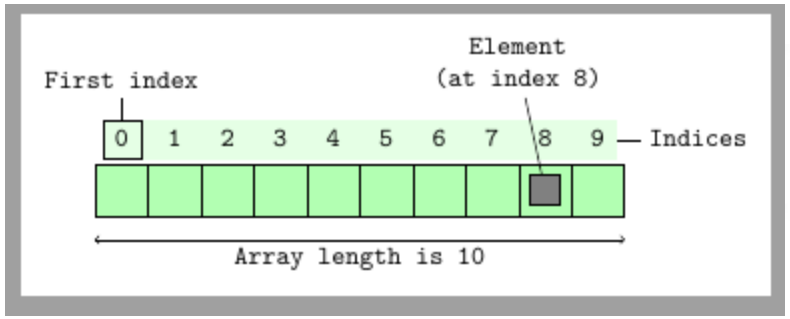
Common Data Structure Operations

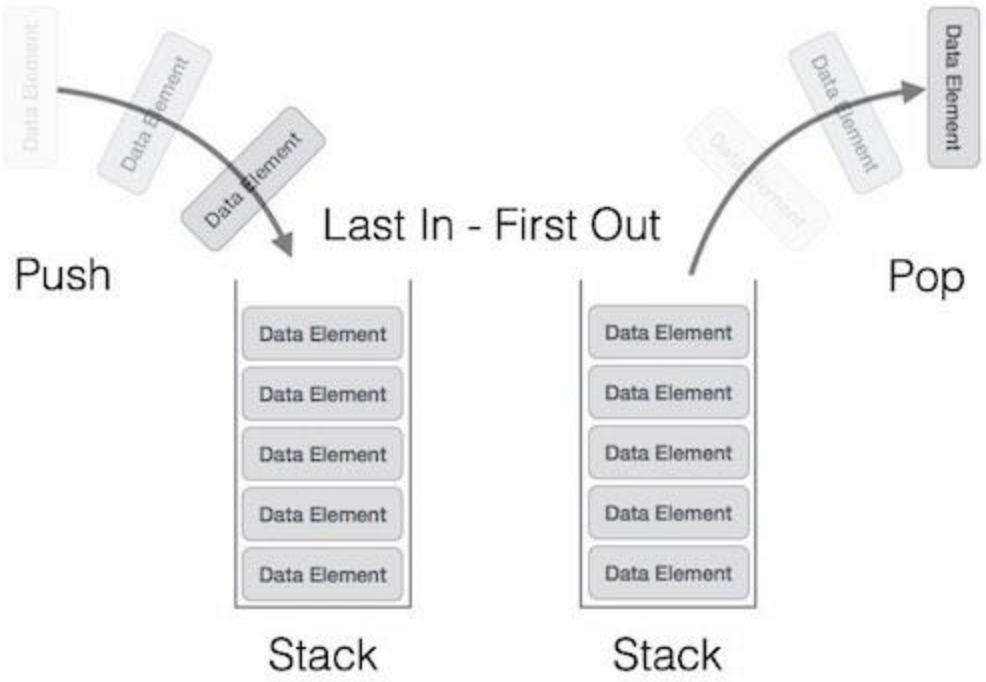
Data Structure	Time Complexity								Space Complexity
	Average				Worst				Worst
	Access	Search	Insertion	Deletion	Access	Search	Insertion	Deletion	
Array	Θ(1)	Θ(n)	Θ(n)	Θ(n)	O(1)	O(n)	O(n)	O(n)	O(n)
Stack	Θ(n)	Θ(n)	Θ(1)	Θ(1)	O(n)	O(n)	O(1)	O(1)	O(n)
Queue	Θ(n)	Θ(n)	Θ(1)	Θ(1)	O(n)	O(n)	O(1)	O(1)	O(n)
Singly-Linked List	Θ(n)	Θ(n)	Θ(1)	Θ(1)	O(n)	O(n)	O(1)	O(1)	O(n)
Doubly-Linked List	Θ(n)	Θ(n)	Θ(1)	Θ(1)	O(n)	O(n)	O(1)	O(1)	O(n)
Skip List	Θ(log(n))	Θ(log(n))	Θ(log(n))	Θ(log(n))	O(n)	O(n)	O(n)	O(n)	O(n log(n))
Hash Table	N/A	Θ(1)	Θ(1)	Θ(1)	N/A	O(n)	O(n)	O(n)	O(n)
Binary Search Tree	Θ(log(n))	Θ(log(n))	Θ(log(n))	Θ(log(n))	O(n)	O(n)	O(n)	O(n)	O(n)
Cartesian Tree	N/A	Θ(log(n))	Θ(log(n))	Θ(log(n))	N/A	O(n)	O(n)	O(n)	O(n)
B-Tree	Θ(log(n))	Θ(log(n))	Θ(log(n))	Θ(log(n))	O(log(n))	O(log(n))	O(log(n))	O(log(n))	O(n)
Red-Black Tree	Θ(log(n))	Θ(log(n))	Θ(log(n))	Θ(log(n))	O(log(n))	O(log(n))	O(log(n))	O(log(n))	O(n)
Splay Tree	N/A	Θ(log(n))	Θ(log(n))	Θ(log(n))	N/A	O(log(n))	O(log(n))	O(log(n))	O(n)
AVL Tree	Θ(log(n))	Θ(log(n))	Θ(log(n))	Θ(log(n))	O(log(n))	O(log(n))	O(log(n))	O(log(n))	O(n)
KD Tree	Θ(log(n))	Θ(log(n))	Θ(log(n))	Θ(log(n))	O(n)	O(n)	O(n)	O(n)	O(n)

Array Sorting Algorithms

Algorithm	Time Complexity			Space Complexity
	Best	Average	Worst	Worst
Quicksort	Ω(n log(n))	Θ(n log(n))	O(n^2)	O(log(n))

Example of array and stack operation





Example of Math Operation

- **Additions is O(n) → linear function, O(n) = n**
- **Subtractions is O(n) → linear function, O(n) = n**
- **Multiplicity is $O(n^2)$ → quadratic function, for example $O(n) = n^2+(2n-1)$**

With:

O(n) is number of operation

n is number of element

For example 10 + 10 can be considered as having 2 elements per component and 100 + 100 can be considered as having 3 elements per component (we compare apple to apple here).

EXAMPLE: Additions operation

10
10
--- +
20 → 2 operations

100
100
------ +
200 → 3 operations

EXAMPLE: MULTIPLICITY

10
10
--------- X
00 → 2 operations
10 → 2 operations
-------- +
100 → 3 operations

Total: 2 + 2 + 3 operations or $2^2 + 3$

Satisfies function $O(n) = n^2+(2n-1)$

100
100
--------- X
000 → 3 operations
000 → 3 operations
100 → 3 operations
-------- +
10000 → 5 operations

Total: 3 + 3 + 3 + 5 operations or $3^2 + 5$

Also satisfies function $O(n) = n^2+(2n-1)$

Quadratic function

Algorithm

DEFINITION:

- **"An algorithm is a well-defined procedure that allows a computer to solve a problem"**
- **"A self-contained step-by-step set of operations to be performed"**
- **"A set of rules that precisely defines a sequence of operations"**
- **Another way to describe an algorithm is a sequence of unambiguous instructions. The use of the term 'unambiguous' indicates that there is no room for subjective interpretation. Every time you ask your computer to carry out the same algorithm, it will do it in exactly the same manner with the exact same result.**

Algorithm: EXAMPles

- **A very simple example of an algorithm would be to find the largest number in an unsorted list of numbers (L).**
- **Step 1: Let variable Largest = L1**
- **Step 2: For each item in the list L:**
- **Step 3: If the item is greater than Largest:**
- **Step 4: Then Largest = the item**
- **Step 5: Return Largest**

ANOTHER EXAMPLE…

Step 1: Start

Step 2: Create a variable to receive the user's email address

Step 3: Clear the variable in case it's not empty

Step 4: Ask the user for an email address

Step 5: Store the response in the variable

Step 6: Check the stored response to see if it is a valid email address

Step 7: Not valid? Go back to Step 3.

Step 8: End

Example in R for Twitter Text Analysis

1. Retrieve tweets
2. Load tweets
3. Convert tweets to a data frame
4. Build a corpus and specify the source to be character vectors
5. Convert corpus to lower case
6. Remove urls
7. Remove anything other than English letters or space
8. Remove punctuations
9. So on …

We are not finished yet…

20. Count frequency of several words at interest

.

.

.

30. Plot
31. Find the association using findAssocs

And more…

PROCEDURE

- **Algorithm can be complex, developers created procedures to make it simpler. For example you can use function MAX(array) to find largest number, similarly you can use max(dat, na.rm=TRUE) in R or Max(Range) in Excel.**

Trade-off in processing complex data analytics

- **The two most common measures are:**

1. ***Time*: how long does the algorithm take to complete.**
2. ***Space*: how much working memory (typically RAM) is needed by the algorithm. This has two aspects: the amount of memory needed by the code, and the amount of memory needed for the data on which the code operates.**

- **For computers whose power is supplied by a battery (e.g. laptops), or for very long/large calculations (e.g. supercomputers), other measures of interest are:**

1. ***Direct power consumption*: power needed directly to operate the computer.**
2. ***Indirect power consumption*: power needed for cooling, lighting, etc.**

Other measurement

- **In some cases other less common measures may also be relevant:**

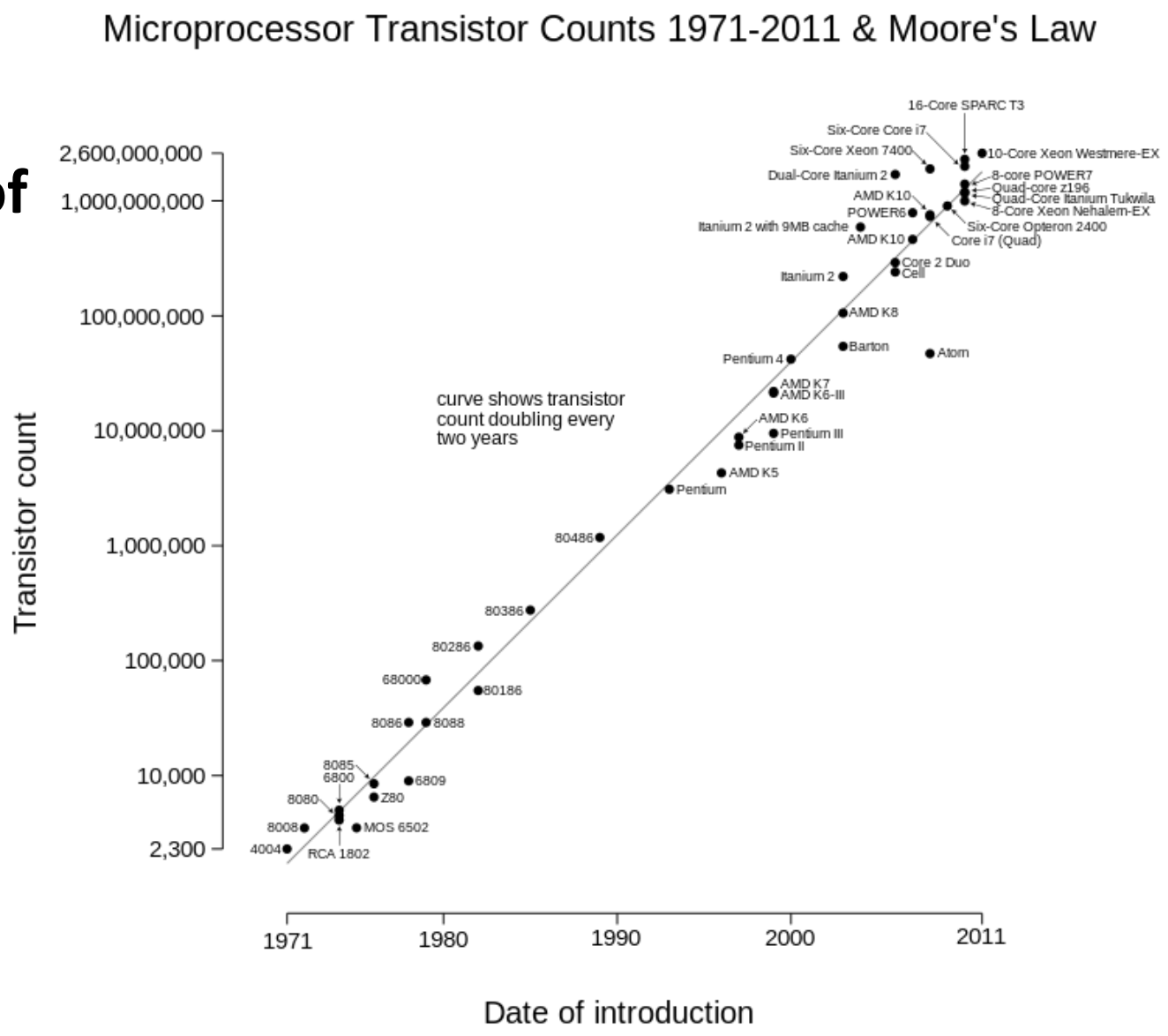
1. ***Transmission size*: bandwidth could be a limiting factor. Data compression can be used to reduce the amount of data to be transmitted. Displaying a picture or image (e.g. Google logo) can result in transmitting tens of thousands of bytes (48K in this case) compared with transmitting six bytes for the text "Google".**
2. ***External space*: space needed on a disk or other external memory device; this could be for temporary storage while the algorithm is being carried out, or it could be long-term storage needed to be carried forward for future reference.**
3. ***Response time*: this is particularly relevant in a real-time application when the computer system must respond quickly to some external event.**
4. ***Total cost of ownership*: particularly if a computer is dedicated to one particular algorithm.**

Exponential in computer technology

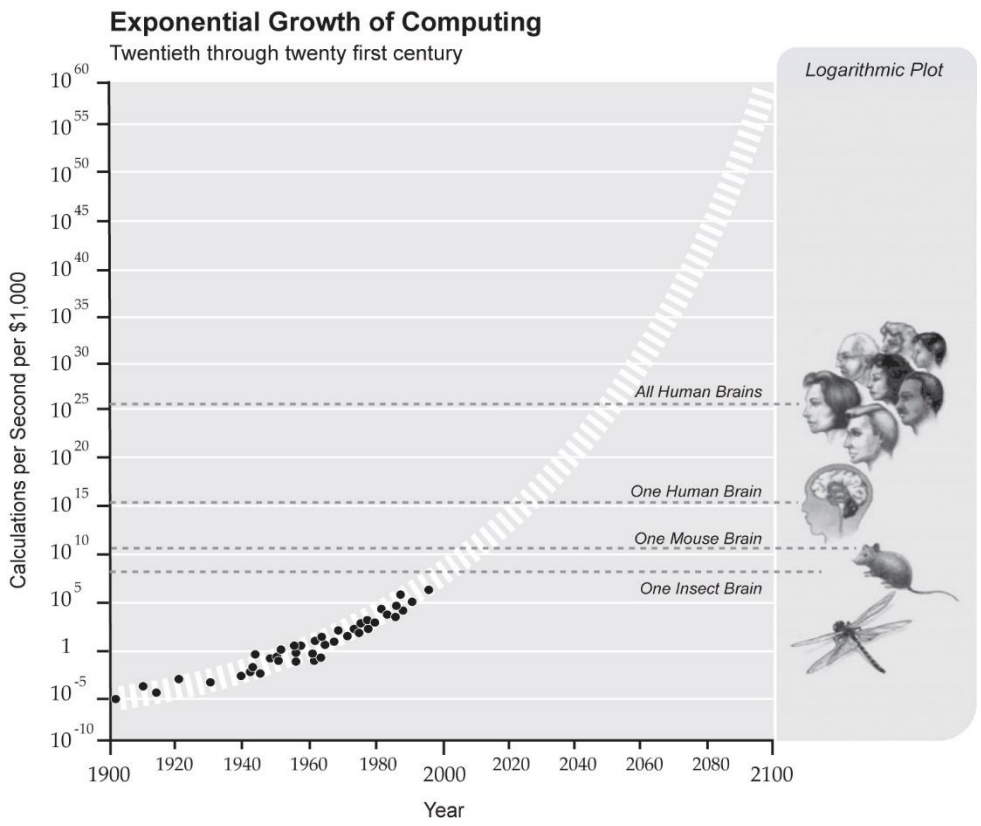
1. Processing power of computers. See also Moore's law and technological singularity. (Under exponential growth, there are no singularities. The singularity here is a metaphor, meant to convey an unimaginable future. The link of this hypothetical concept with exponential growth is most vocally made by transhumanist Ray Kurzweil.)
2. In computational complexity theory, computer algorithms of exponential complexity require an exponentially increasing amount of resources (e.g. time, computer memory) for only a constant increase in problem size. So for an algorithm of time complexity 2^x, if a problem of size $x = 10$ requires 10 seconds to complete, and a problem of size $x = 11$ requires 20 seconds, then a problem of size $x = 12$ will require 40 seconds. This kind of algorithm typically becomes unusable at very small problem sizes, often between 30 and 100 items (most computer algorithms need to be able to solve much larger problems, up to tens of thousands or even millions of items in reasonable times, something that would be physically impossible with an exponential algorithm). Also, the effects of Moore's Law do not help the situation much because doubling processor speed merely allows you to increase the problem size by a constant. E.g. if a slow processor can solve problems of size x in time t, then a processor twice as fast could only solve problems of size x+constant in the same time t. So exponentially complex algorithms are most often impractical, and the search for more efficient algorithms is one of the central goals of computer science today.
3. Internet traffic growth

Moore's law

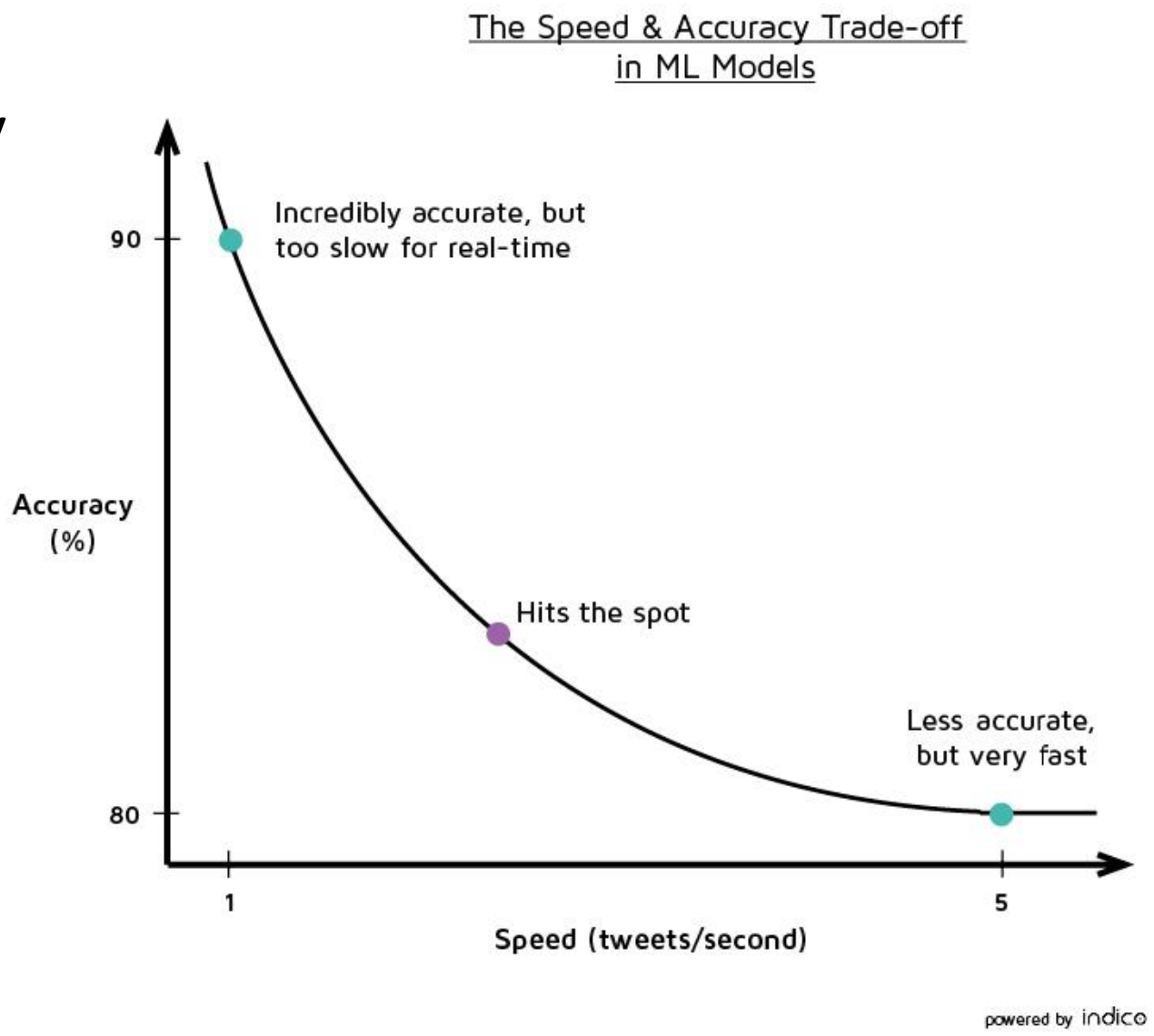
- **Moore's law (/mɔərz.ˈlɔː/) is the observation that the number of transistors in a dense integrated circuit doubles approximately every two years.**



Computational POWER



Choose what's best for you (or you may say Optimization)



Level of optimization

1. **Design level**
2. **Algorithms and data structures**
3. **Source code level**
4. **Build level**
5. **Compile level**
6. **Assembly level**
7. **Run time**

Our interest for this course

Strength reduction

- **Computational tasks can be performed in several different ways with varying efficiency. A more efficient version with equivalent functionality is known as a strength reduction.**
- **For example, consider the following C code snippet whose intention is to obtain the sum of all integers from 1 to N:**
- **int i, sum = 0;**
- **for (i = 1; i <= N; ++i) {**
- ** sum += i;**
- **}**
- **printf("sum: %d\n", sum);**
- **This code can (assuming no arithmetic overflow) be rewritten using a mathematical formula like:**
- **int sum = N * (1 + N) / 2;**
- **printf("sum: %d\n", sum);**

Strength Reduction should…

1. **Minimize space / size**
2. **Minimize time**

Take examples in apps optimization. Optimized apps have characteristics:

1. **Run faster (means more efficient)**
2. **Take less space (Before optimization: 1GB, after optimization: 0.9GB)**
3. **Preferably take less RAM space**

These characteristics also apply to algorithm.

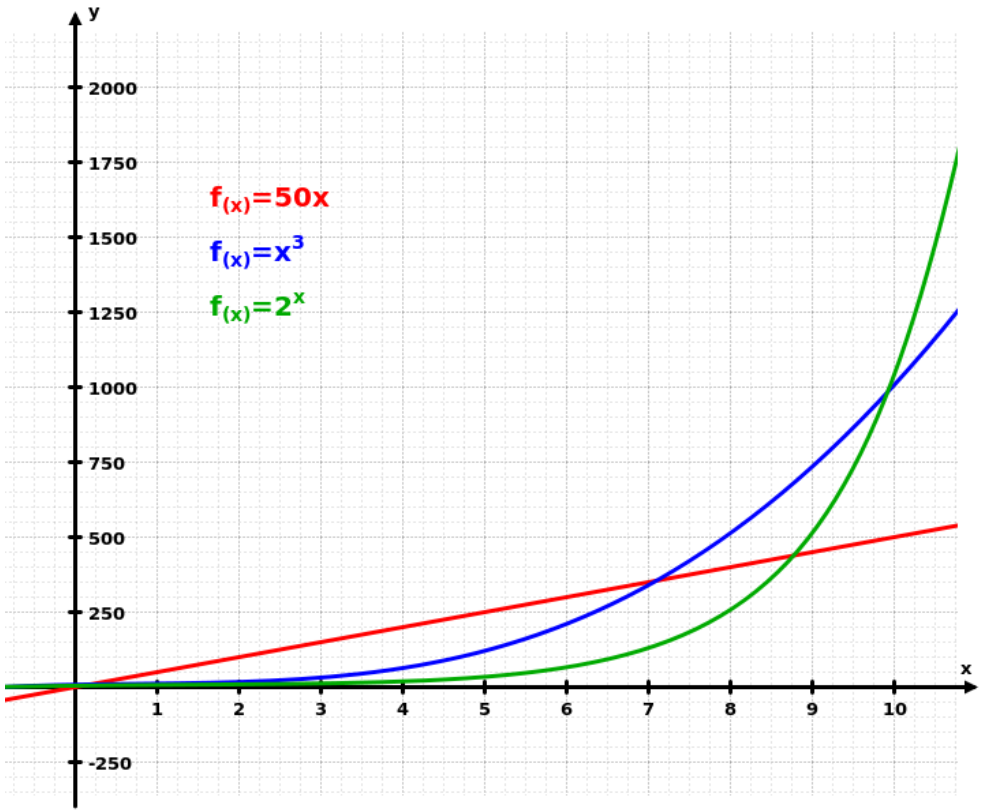
Things grow fast: exponentially

- **Exponential growth is a phenomenon that occurs when the growth rate of the value of a mathematical function is proportional to the function's current value, resulting in its growth with time being an exponential function.**

Green: Exponential growth

Red: Linear growth

Blue: Cubic growth



Borrow best practices from management knowledge

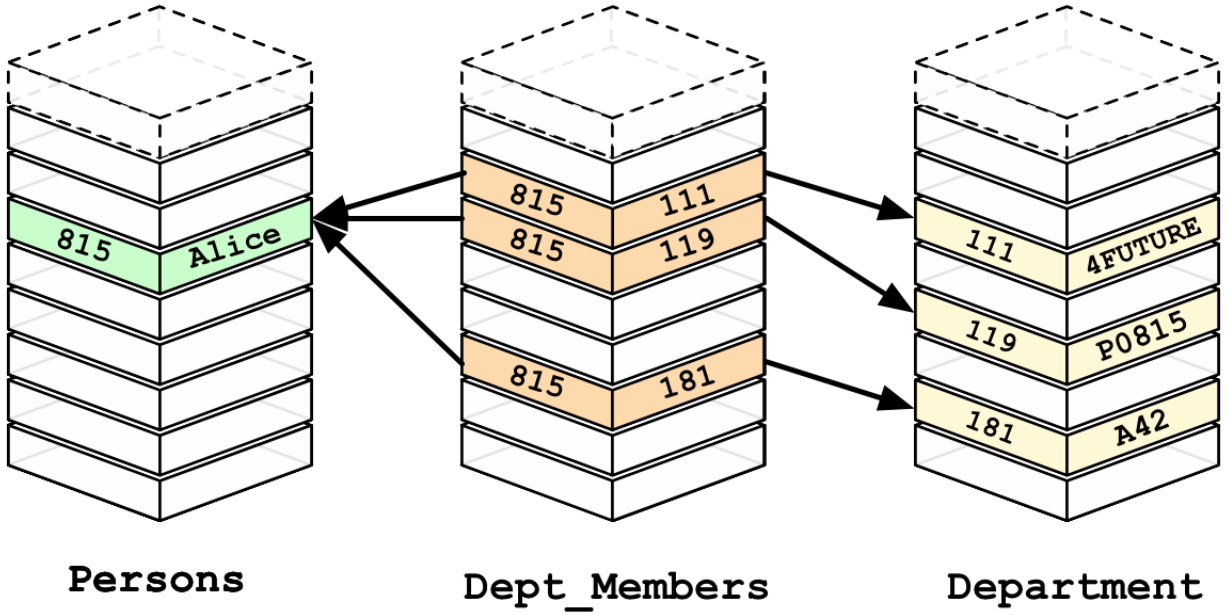
How To Reduce Complexity In Five Simple Steps

1. **Clear the underbrush, get rid of ambiguous rules and low-value activities, time-wasters**
2. **Clear perspective, focus on specific goals**
3. **Prioritize most important things**
4. **Take shortest path by eliminating loops, redundancies, and also create things leaner**
5. **Reduce levels**

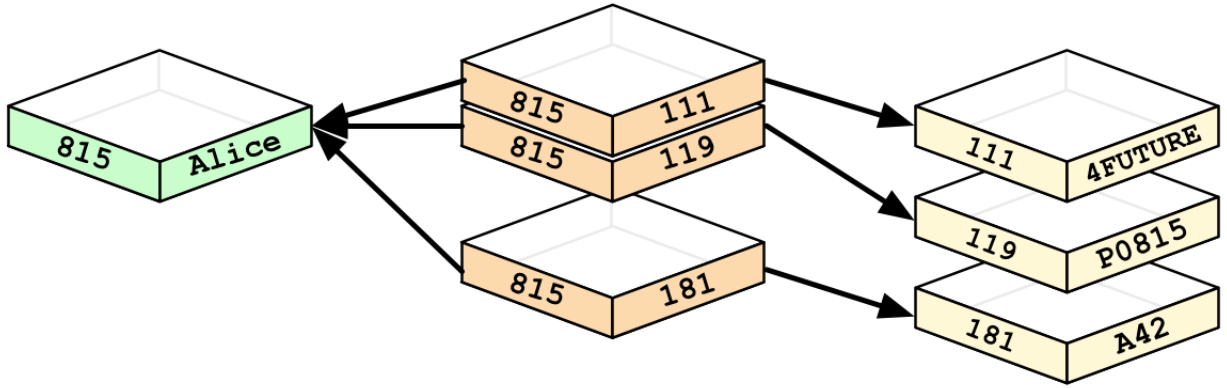
Using graph database for complex network/relationship intensive data

- **GRAPH DATABASE**
- **In computing, a graph database is a database that uses graph structures for semantic queries with nodes, edges and properties to represent and store data. A key concept of the system is the *graph* (or *edge* or *relationship*), which directly relates data items in the store. The relationships allow data in the store to be linked together directly, and in most cases retrieved with a single operation.**
- **This contrasts with conventional relational databases, where links between data are stored in the data itself, and queries search for this data within the store and use the JOIN concept to collect the related data. Graph databases, by design, allow simple and rapid retrieval of complex hierarchical structures that are difficult to model in relational systems. Graph databases are similar to 1970s network-model databases in that both represent general graphs, but network-model databases operate at a lower level of abstraction[1] and lack easy traversal over a chain of edges.[2]**

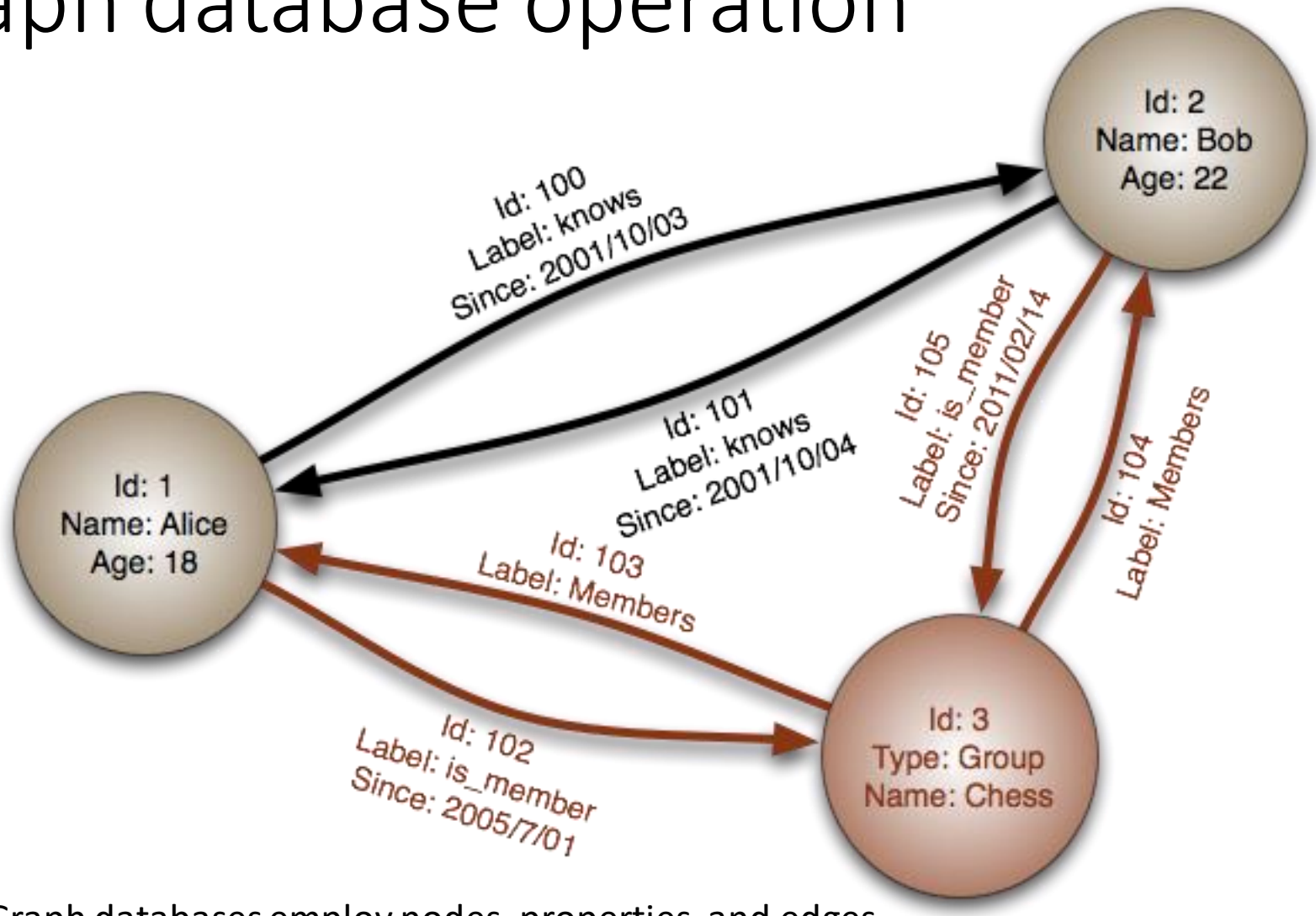
Your RDBMS typical storage



Graph database approach



Typical graph database operation



Graph databases employ nodes, properties, and edges.

Popular graph databases softwares

db-engines.com/en/ranking/graph+dbms

- Multivalue DBMS
- Native XML DBMS
- Wide column stores
- Content stores
- Event Stores
- Navigational DBMS

Special reports

- Ranking by database model
- Open source vs. commercial

Featured Products

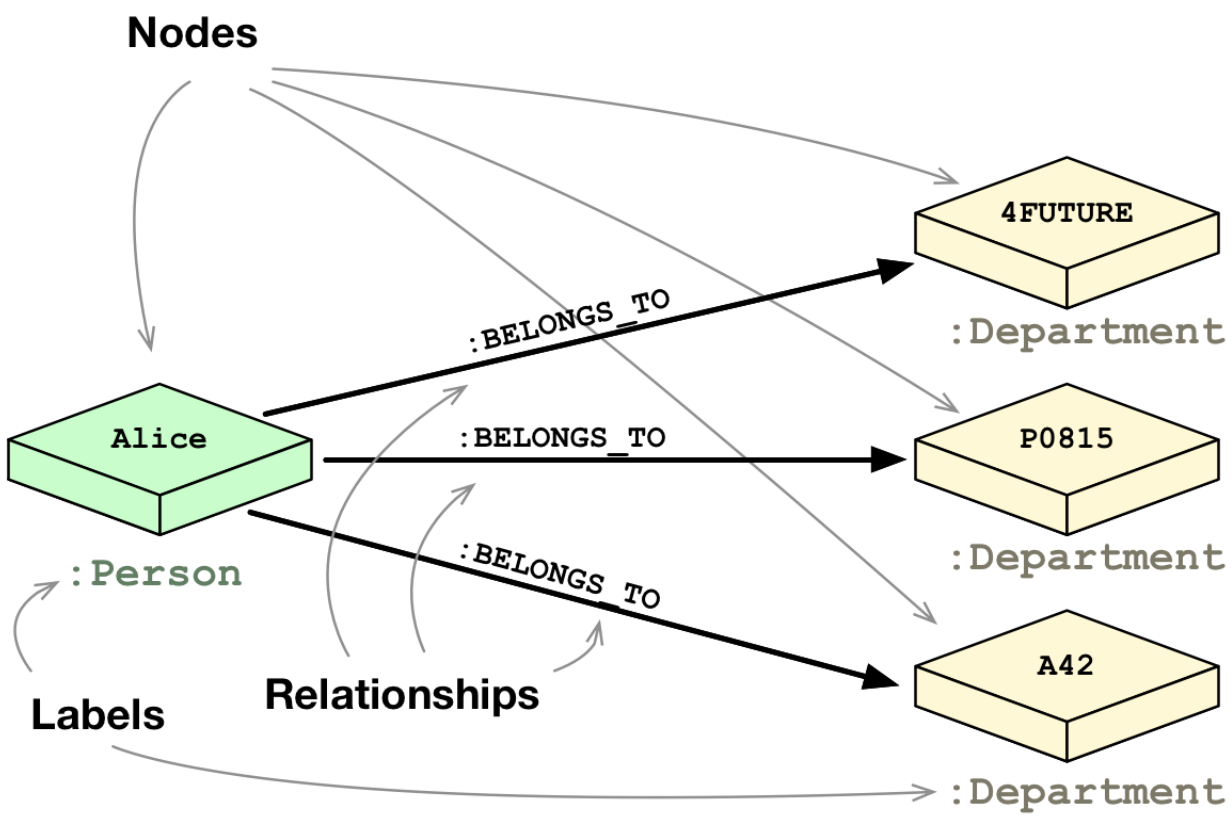
Franz Inc.
AllegroGraph
Semantic Graph Database Leader - Add The Power of Cognitive Probability Graphs - Unique Graph Visualizations - Free Download

21 systems in ranking, September 2016

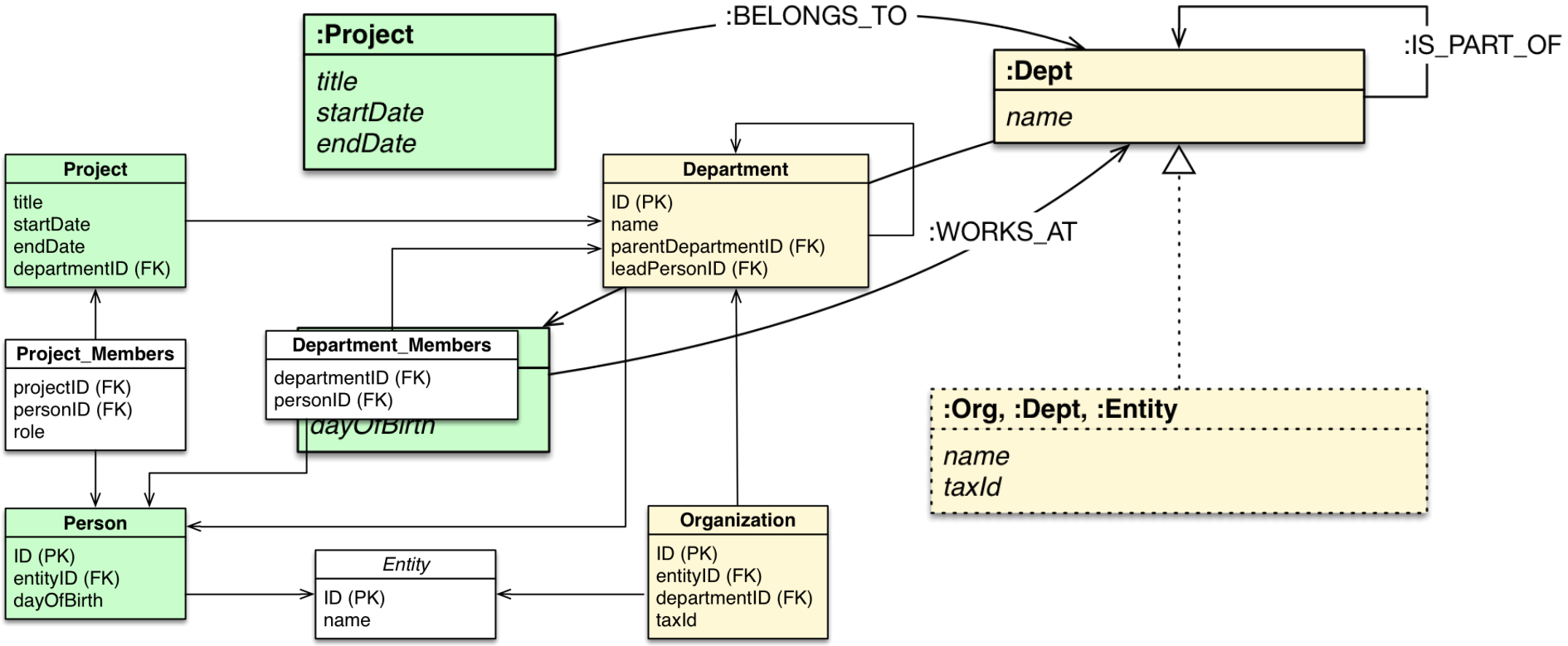
Rank Sep 2016	Rank Aug 2016	Rank Sep 2015	DBMS	Database Model	Score Sep 2016	Score Aug 2016	Score Sep 2015
1.	1.	1.	Neo4j	Graph DBMS	36.37	+0.80	+2.83
2.	2.	2.	OrientDB	Multi-model	6.40	+0.43	+1.80
3.	3.	3.	Titan	Graph DBMS	5.06	+0.17	+0.55
4.	4.	4.	Virtuoso	Multi-model	2.50	+0.10	-0.71
5.	5.	5.	ArangoDB	Multi-model	2.13	+0.20	+0.77
6.	6.	6.	Giraph	Graph DBMS	1.12	+0.17	+0.07
7.	7.	↑ 8.	Stardog	Multi-model	0.60	+0.06	+0.18
8.	8.	↓ 7.	AllegroGraph	Multi-model	0.51	+0.05	-0.30
9.	9.	↑ 10.	Sqrrl	Multi-model	0.36	+0.10	+0.02
10.	↑ 12.		Dgraph	Graph DBMS	0.20	+0.06	
11.	↓ 10.	↓ 9.	InfiniteGraph	Graph DBMS	0.19	-0.00	-0.21
12.	↓ 11.	↑ 17.	GraphDB	Multi-model	0.16	-0.01	+0.10
13.	↑ 15.	↑ 19.	Blazegraph	Multi-model	0.16	+0.07	+0.15
14.	↓ 13.	↓ 13.	FlockDB	Graph DBMS	0.12	-0.00	-0.06
15.	↓ 14.	↓ 14.	InfoGrid	Graph DBMS	0.09	-0.00	-0.06
16.	↑ 18.	↓ 15.	GlobalsDB	Multi-model	0.08	+0.04	-0.05
17.	↓ 16.	↓ 12.	HyperGraphDB	Graph DBMS	0.08	-0.00	-0.11
18.	↓ 17.	↓ 11.	Sparksee	Graph DBMS	0.04	-0.03	-0.22
19.	19.	↓ 18.	GraphBase	Graph DBMS	0.03	+0.00	-0.03
20.	↑ 21.	20.	Amisa Server	Multi-model	0.00	±0.00	±0.00
20.	20.	↓ 16.	VelocityGraph	Graph DBMS	0.00	-0.02	-0.09

Source: db-engines.com

neo4J data model



Rdbms vs graph dbms: data structure



Rdbms vs graph dbms: query

- **SQL statement**
- **SELECT name FROM Person LEFT JOIN Person_Department ON Person.Id = Person_Department.PersonId LEFT JOIN Department ON Department.Id = Person_Department.DepartmentId WHERE Department.name = "IT Department"**

NoSQL statement: Using Cypher in Neo4J

```
MATCH (p:Person)<-[:EMPLOYEE]-(d:Department)
WHERE d.name = "IT Department"
RETURN p.name
```

Wrap up: strategy in managing big data analytics

- **Utilizing best practices to gain valuable insight from big data by employing these concepts:**

1. **Data usability**
2. **Data integration into key processes**
3. **Actionable insight that improve decision making processes**
4. **Data share**
5. **Best tools**
6. **Scalability and Speed**
7. **Reduce complexity**

Exercise (tentative)

1. **Identify complex systems in daily life that can be managed by computational system (eg. Information System, DSS, ERP, etc.). In class.**
2. **Try to differentiate between 4 type of problem contexts (simple/obvious, complicated, complex, chaos) for different systems. In Class.**
3. **Search for a case study of a company's strategy on managing big data analytics (may use your prior case study). You may give your suggestions. In class or homework.**

Assessment Metrics:

1. **Number of component in the system (eg. Stakeholders, subsystem, softwares, storage, etc.) to identify size or space**
2. **Length of time (eg. Data timelime, process length, etc.)**
3. **Number of suggestions related to points in "Strategy in Managing Big Data Analytics"**

Sources

1. P. Ferreira, "Tracing Complexity Theory"
2. Angles, Renzo; Gutierrez, Claudio (1 Feb 2008). "Survey of graph database models" (PDF). ACM Computing Surveys. Association for Computing Machinery.
3. Silberschatz, Avi (28 January 2010). Database System Concepts, Sixth Edition
4. Frost Sullivan, "Reducing Information Technology Complexities and Costs For Healthcare Organizations", retrieved on September 2016 from https://www.emc.com/collateral/analyst-reports/frost-sullivan-reducing-information-technology-complexities-ar.pdf
5. Julia Wester, "Understanding the Cynefin framework – a basic intro", retrieved on September 2016 from http://www.everydaykanban.com/2013/09/29/understanding-the-cynefin-framework/