Digital Transformation using chunks as a simple abstraction above triples and property graphs

Dave Raggett, W3C/ERCIM

SEMANTiCS 2021 Workshop on squaring the circle on graphs

6th September 2021

This work has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement № 957406 (TERMINET) Next Generation Smart Interconnected IoT

Digital Transformation

- Digital transformation is the reshaping of businesses for the digital age
- Businesses are seeking to become more efficient, more transparent and more agile through the exploitation of digital technologies throughout the enterprise
- Businesses need to integrate heterogenous information systems and formats
  - including SQL/RDBMS, Spreadsheets, CSV files, XML, Property Graphs, Linked Data, PDF files, etc.
- Change is inevitable and on-going
- Different groups in an enterprise use different ways of talking about things
  - Along with different tools for this
- Software development is generally speaking expensive, error prone and time consuming
  - Need to commit upfront to data models
  - But requirements evolve over time
  - Updating software and databases is hard
- Need for incremental rather than revolutionary change
  - This includes enterprise personnel!
- Enterprise knowledge graphs offer greater flexibility than SQL/RDBMS databases
  - Data models can be added when needed
  - Capturing knowledge otherwise implicit
  - *A knowledge graph can answer what it knows, and also how and why it knows it**

* WTF is a knowledge graph, Jo Stichbury, 2017

Web-based access to knowledge graphs

- □ Enterprise knowledge graphs
  - Integrated store for enterprise systems
  - Contain data, data models, semantics and metadata
  - Federated across organisational units and geographic regions as required
  - Can be searched in a style closer to natural language, analogous to smart Web search
  - Can include on-demand & proactive services relating to the data, and to the roles of different users

- □ Some requirements
  - Level of detail control for complex diagrams: hierarchical structure, search queries and views
  - Machine interpretable serialisation format for diagrams
  - Revision control – keeping track of changes - what, why, and who
  - Re-use of master and reference data models
  - Validation against organisational rules including who can do what
  - Scaling to big data

Web-based Tools

- Today it is common practice to use diagrams during the design stage
  - Mindmap, UML and ER diagrams
- But diagrams are standalone and not integrated into application lifecycle
- Opportunities for Web-based design solutions, dynamically integrated with enterprise knowledge graphs
- Use of diagrams for queries and reporting
- View/edit diagrams in web browser with live synchronisation for editing by distributed teams
- Diagrams saved and centrally managed in the cloud
- Integration as part of enterprise-wide knowledge graph and business processes
- The diagramming tools need to be reversible: diagram to knowledge graph, and knowledge graph to diagram

From Excel to Knowledge Graphs

- Spreadsheets are ubiquitous, but hard for businesses to manage
- Moreover, they tend to grow to the point where they become difficult to maintain
- It is time to wean users to a new generation of web-based spreadsheets where cells are connected live with enterprise knowledge graphs
- Meaningful names for cells rather than letters and numbers
- Need for collaborative tools that help users to migrate from Excel to web-based knowledge sheets
- Expert system that recognises common design patterns for spreadsheets
- Learns new patterns from experience across spreadsheets
- Alignment with existing enterprise knowledge graph concepts where feasible

Beyond low-code/no-code

- □ Low-code/no-code is claimed to be the future of software development*
  - Low-code/no-code development platforms are types of visual software development environments that allow enterprise developers to drag and drop application components, connect them together and create mobile or web apps
  - They also enable business analysts, office administrators, small-business owners and others who are not software developers to build and test applications
  - These people can create applications with little to no knowledge of traditional programming languages, machine code or the development work behind the platform's configurable components
  - Reduced dependency on IT department
  - Faster access to working solutions

- □ But why develop applications when you can ask smart cognitive agents to solve problems for you?
  - Cognitive agents can figure out how to implement tasks in terms of composing and configuring services
  - Users describe what they want, agent clarifies this through multimodal dialogue
    - Natural language and diagrams
    - Extension of query by example
  - Agent then uses mix of declarative and procedural knowledge to compose and execute a solution
    - Agent uses human-like reasoning
  - Declarative knowledge expressed using enterprise knowledge graph
    - Including business processes
  - Procedural knowledge exposed as services with machine interpretable descriptions

* Courtesy of Stephen J. Bigelow, TechTarget, March 2021

Chunks and Human Memory

Graph databases inspired by the cognitive sciences

- Chunks are typed sets of properties whose values are literals, references to other chunks, or comma separated sequences thereof *
- Convenient syntax that is simpler than JSON and JSON-LD

```
dog dog1 {
   name "fido"
   age 4
}
```

```
dog dog1 {name "fido"; age 4}
```

- Use whitespace as you please, with semicolon or newline as punctuation
- Chunk ID is optional and automatically assigned if missing
- Even simpler syntax for binary relationships
- Chunk rules for reasoning (see later slide)

- Chunks can be separated into named contexts for things that are only true in a given context
  - Hypothetical situations and plans
  - Statements about statements
  - Past episodes and fictional narratives
  - *Equivalent to RDF named graphs*
- Graph API for implementing graph algorithms
- Chunks are associated with activation levels that are boosted by accesses and updates, and decay over time
  - Spreading activation for boosting related memories
- Chunk retrieval is stochastic, based on activation levels
  - Can be disabled when appropriate
- This mimics human memory and ensures that useful memories are more likely to be retrieved based on past experience
  - Ebbinghaus “forgetting curve”

* A minimalist version of chunks limits properties to just the names of other chunks. *Type* is used for loose grouping and may imply an ontological constraint.

Chunks and RDF

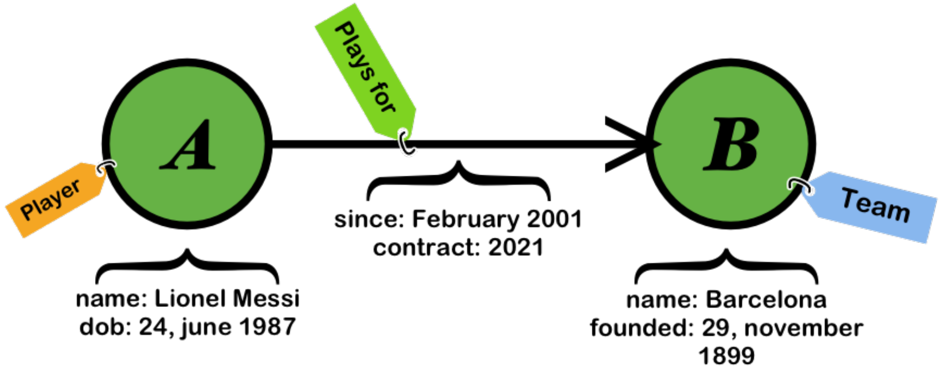
- Chunks correspond to n-ary relations in RDF involving a common subject node
- Chunks are easier to work with than other RDF serialisations
- Chunks come with a simple rule language for procedural knowledge
- Chunks are aimed at informal, human-like, reasoning, NOT formal semantics and proofs
  - See later slide on reasoning

- Mappings with RDF similar to JSON-LD
  - @rdfmap to map names to RDF URIs, either in place, or by reference to external definitions
    - Can be used to map chunk types, property names, and named values
    - @context used for named contexts
  - @base to set default URI
  - @prefix to declare URI prefixes

```
@prefix p1 { ex: http://example.com/ns/ }
@rdfmap {
   @prefix p1
   dog ex:dog
   cat ex:cat
}
```

Literals can be Booleans, numbers, strings, URIs, and dates which implicitly expand into chunks with properties for year, month and day

Chunks and Property Graphs



- ❑ Property Graphs
  - ▪ Both graph nodes and links can have properties
- ❑ Chunks can be used for both nodes and links
  - ▪ Chunk properties can be promoted to separate relationships as and when needed

```
// Neo4J CYPHER query example*

CREATE  (m:Player{name:"Lionel Messi"}),
  (b:Team{name:"Barcelona"})
WITH m, b
CREATE (m)-[p:PlaysFor]->(b)
SET p.since=2001
RETURN m, p, b
```

```
# PG Nodes
Player p413 {
   name "Lionel Messi"
   dob 1987-06-24
   # PlaysFor t87
}
Team t87 {
   name "Barcelona"
   founded 1899-11-29
}
```

```
# PG Link
PlaysFor r45 {
   @subject p413
   @object t87
    since 2001-02
    contract 2021
}
```

* From: https://neo4j.com/graphgists/introduction-to-neo4j-2/

Human-like Reasoning

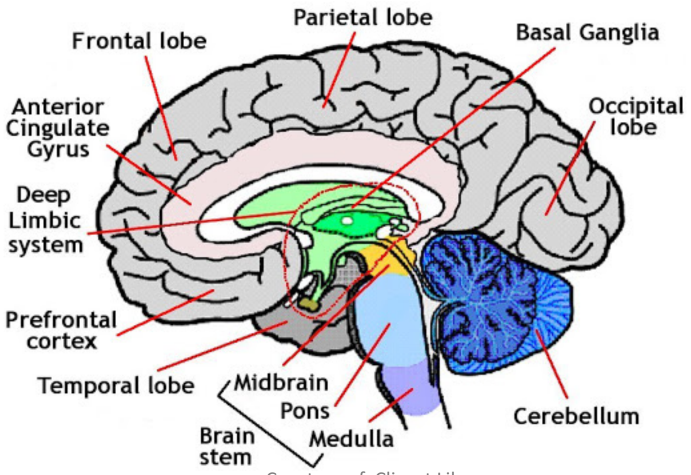
- Humans do not reason with logical proofs or Bayesian statistics, but rather with examples and analogies
  - Philip Johnson-Laird’s *“As We Think”*
- Informal reasoning with natural language semantics
- We are good at different kinds of reasoning beyond deduction
  - Causal relationships
  - Abduction and intents
  - Use of metaphors and analogies
  - Planning and imagination

- Unsupervised learning has been very successful (eg GPT-3)
  - Learns to predict following text
  - But is weak in respect to reasoning
  - Requires huge models and vast corpora
  - Not very human-like!
- Big opportunity for weakly supervised incremental learning
  - Text corpora, WordNet, taxonomies
- Plus learning in the classroom and playground
  - Mimic how children learn
  - Ground semantics in interaction

Note contrast between graphs with explicit symbols vs distributed representations with vector spaces, e.g. neural networks. Distributed representations may be more effective, e.g. for machine learning with sparse statistics, and non-symbolic learned features, and have been very successful for machine translation, speech recognition and web search.

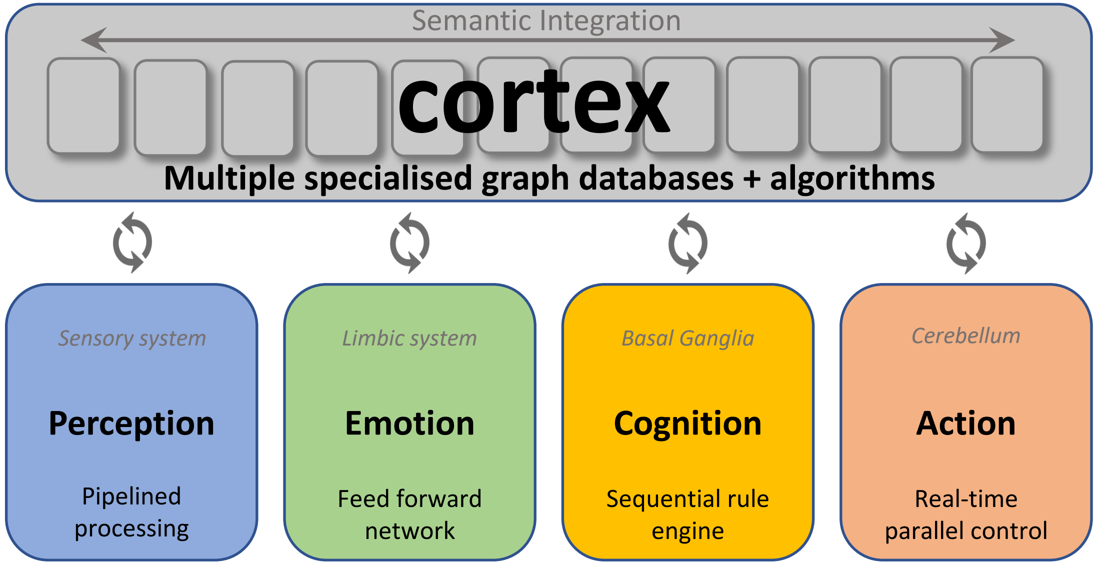
Cognitive Architecture for human-like AI

Basis for next generation graph database systems



Courtesy of Clipart Library

Cognitive Architecture with multiple cognitive circuits loosely equivalent to shared blackboard



Semantic Integration

cortex

Multiple specialised graph databases + algorithms

Sensory system

Perception

Pipelined processing

Limbic system

Emotion

Feed forward network

Basal Ganglia

Cognition

Sequential rule engine

Cerebellum

Action

Real-time parallel control

Anterior temporal lobe as hub for integration across senses

- **Perception** interprets sensory data and places the resulting models into the cortex. Cognitive rules can set the context for perception, and direct attention as needed. Events are signalled by queuing chunks to cognitive buffers to trigger rules describing the appropriate behaviour. A prioritised first-in first-out queue is used to avoid missing closely spaced events.

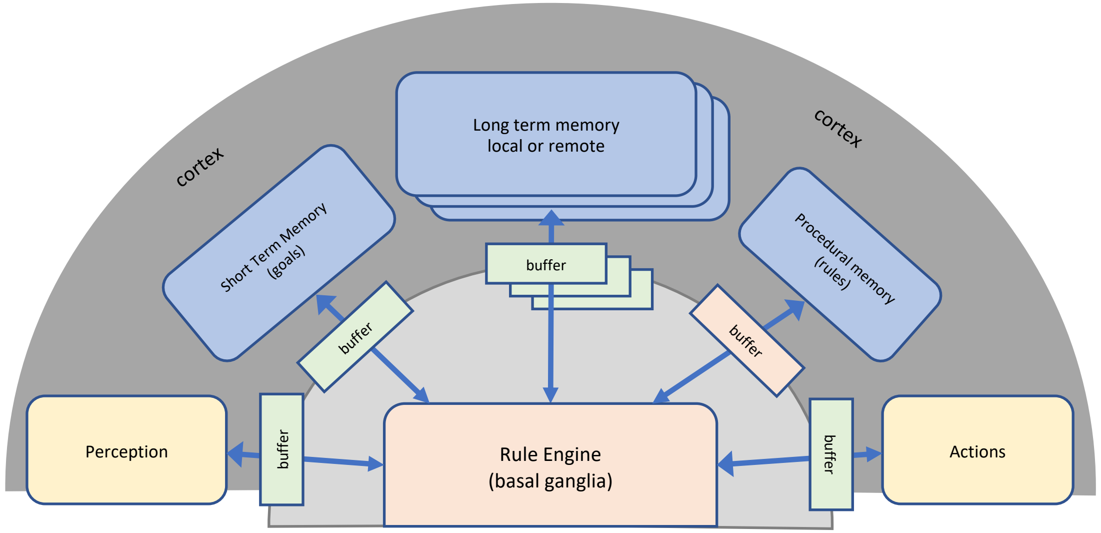
- **Emotion** is about cognitive control, prioritising what's important and learning from the past. The limbic system provides rapid assessment of situations without the delays incurred in deliberative thought. This is sometimes referred to as System 1 vs System 2.

- **Cognition** is slower and more deliberate thought, involving sequential execution of rules to carry out particular tasks, including the means to invoke graph algorithms in the cortex, and to invoke operations involving other cognitive systems. Thought can be expressed at many different levels of abstraction.

- **Action** is about carrying out actions initiated under conscious control, leaving the mind free to work on other things. An example is playing a musical instrument where muscle memory is needed to control your finger placements as thinking explicitly about each finger would be far too slow. The cerebellum coordinates muscle activation guided by perception.

Chunk Rules for System 2 Thinking

- Simple condition-action rules
  - Inspired by ACT-R *and* HTTP*
- Operate on cognitive buffers holding single chunks
  - Conditions use pattern matching
- Asynchronous actions, e.g.
  - Update buffers directly
  - Recall from database
  - Create and update chunks
  - Invoke graph algorithm
  - Invoke external action
- Open source implementation and suite of web-based demos†
  - [W3C Cognitive AI Community Group](#)



Sequential rule engine where buffers correspond to stable concurrent firing patterns of bundles of nerve fibres connecting to different parts of the cortex. The brain runs in parallel locally, and sequentially long distance. Related to Bernard Baars' Global Workspace theory of consciousness.

* Use of HTTP method names for asynchronous operations on graph data

† Chunks and rules specification as W3C Community Group Report

Questions?

Don’t get stuck in a silo, think across disciplines!