SemMemDB

Contextualized Semantic Memory using Spreading Activation

Jay Ricco

riccoj@wit.edu

Wentworth Institute of Technology

Review of Semantic Memory

Fact-based long term memory store

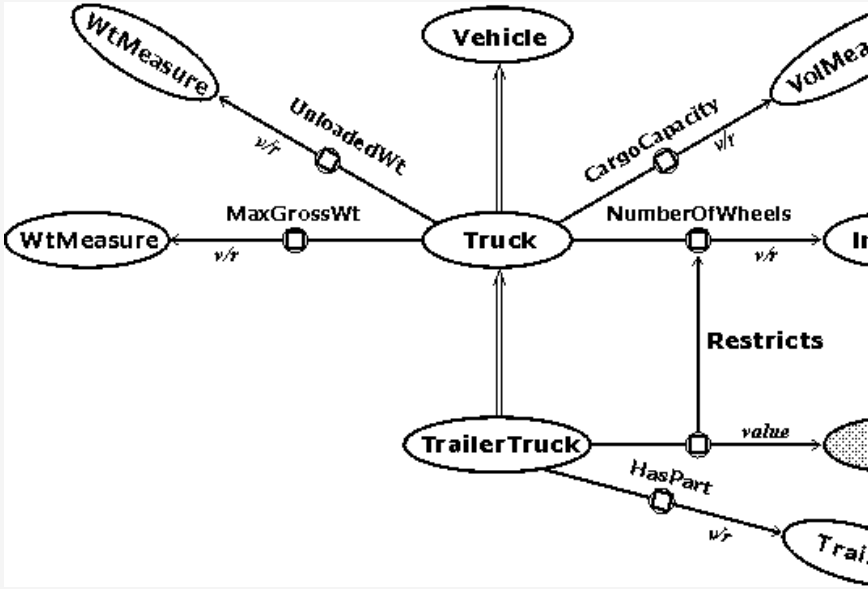
- Songs by a certain artist
- How many wheels a common car has

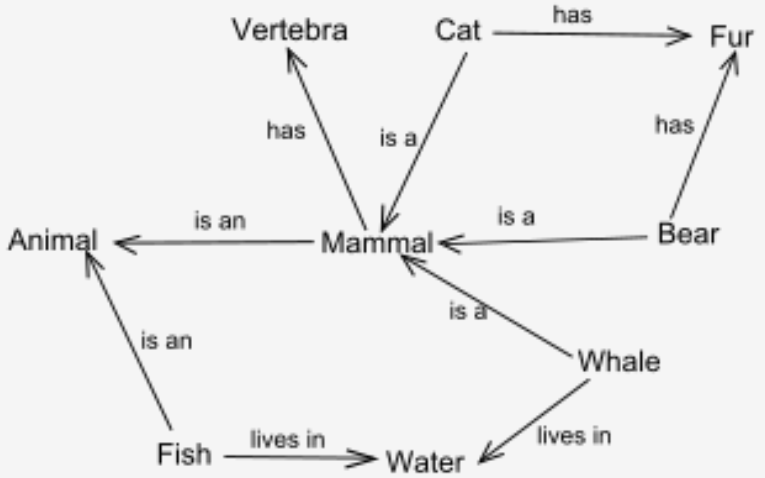
Memory elements stored in a digraph structure

Inserted data is placed in graph structure based on relations as execution continues

Retrieval occurs as either a direct lookup or query-based retrieval

- Query-based retrieval can lead to some issues with both efficiency and accuracy (ambiguity).





Review of Spreading Activation

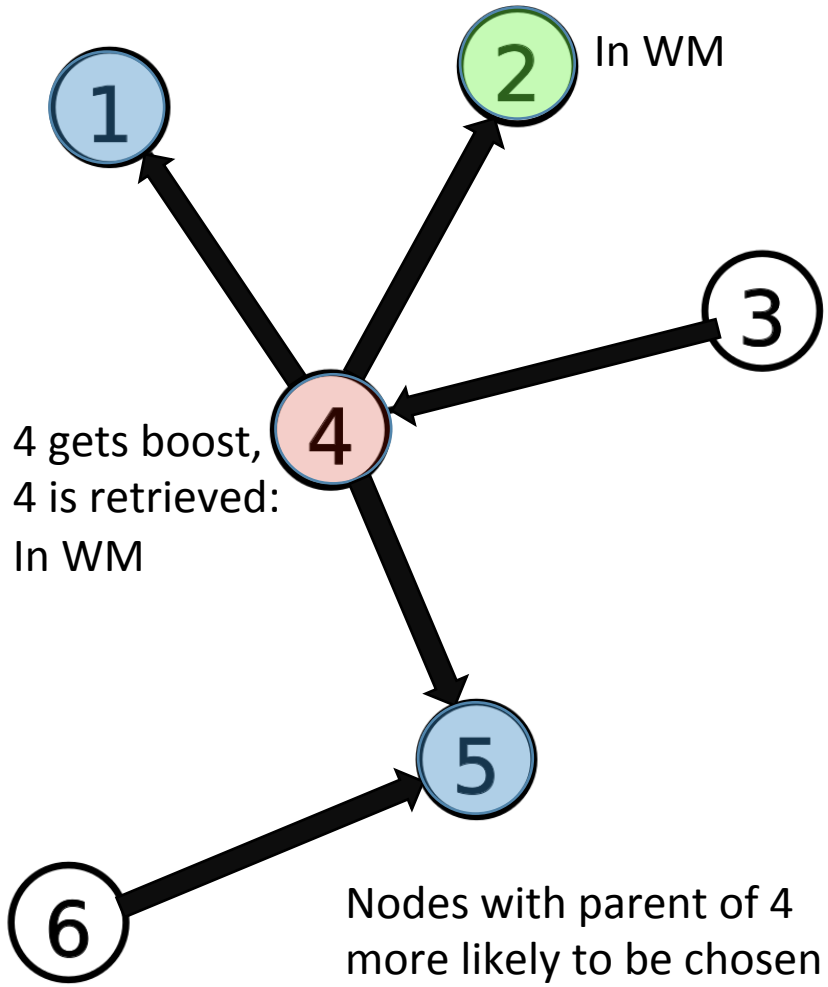
lows context to affect SMEM retrieval
- Word-sense disambiguation
- Conversational topics

eneficial for query-based lookup to crease chances of correct retrieval

ements in working memory provide e "context"

ossible nodes become weighted ased on # of edges between emselves & WME's

he greater the number of relations etween the node & the WME, the rger the probability of retrieval is



SemMemDB (FLAIRS-27 2014)

reated by:

- Yang Chen – University of Florida
- Milenko Petrovic & Micah Clark - Florida Institute for Human & Machine Cognition

atabase implementation of semantic memory

esigned to exploit heuristically optimized query engines of RDBMS's for ssociative information retrieval

- Speed
- SQL
- Massive amounts of data

emantic network stored as a table of symbolic triples:

$$Subject - Predicate\,(relation) - Object$$

SemMemDB - Retrieval

etrieval in SemMemDB is based on ACT-R's base-level activation, with the ddition of spreading.

- Non-probabilistic graph traversal

$$RetrievalScore_i = \ln\left(\sum_{k=1}^{n} t_k^{-d}\right) + \sum_{j \in Q} weight_j * \left(S - \ln\left(1 + degree_j / edges_{ji}\right)\right)$$

Base-level Activation

Spreading Activation

i=subject node in dataset
n=number of presentations of node i
t_k =time since k^{th} presentation of i
d=rate of activation decay

j=object node in dataset
Q is the contextual element source
S=constant parameter
$edges_{ji}$ =relations between j and i

SemMemDB – Weak Points

aper contains efficiency testing only relating dataset size and query size to

oes not provide time data on key elements of retrieval score calculation

- Connectivity of nodes
- History size

heir implementation consists of a feed-forward spread

- Spreading from parent node to child node

id not implement materialized views – used tables/view hybrid instead

- Did not provide data on re-indexing time for dataset inserts
  - If not properly indexed calculation time will increase

oes not incorporate hard constraints (only allows for look-up based on cont
ements)

ot integrated with cognitive architecture

SemMemDB – My Work

urrently have a working implementation of SemMemDB with Postgres
- Toy datasets return accurate spreads from queries of “context” nodes

ur implementation contains a feed-backward spread and materialized views

Future Work:

Comprehensive testing and analysis of timing
- Calculation
- Re-indexing/MV refresh

Drop Postgres as RDBMS and produce a standalone C++ implementation
- No need to connect to DB server/use interface drivers

Optimize algorithms to efficiently be able to retrieve resultant nodes in as lit
time as possible

Golden Nuggets and Coal

Our implementation is currently functional and does produce an accurate spread based on trials with toy datasets

We have a good path-to-goal for our research, and should achieve bigger and better in the coming weeks!

- No official empirical data!