



Balanced Large Scale Knowledge Matching Using LSH Forest

1st International KEYSTONE Conference

IKC 2015

Coimbra Portugal,
8-9 September 2015



Michael Cochez *

Vagan Terziyan *

Vadim Ermolayev **



* Industrial Ontologies Group,
University of Jyväskylä (Finland),
michael.cochez@jyu.fi
vagan.terziyan@jyu.fi

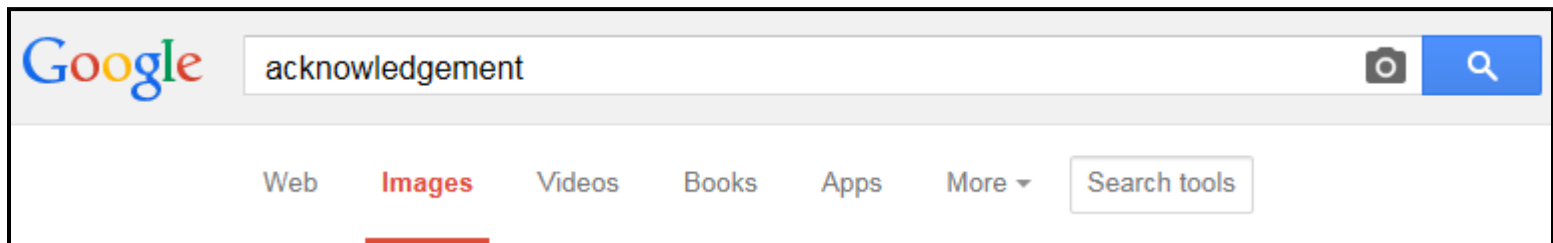


** Zaporizhzhya National
University, (Ukraine),
vadim@ermolayev.com



In Grateful Appreciation

- We are grateful to the anonymous photographers and artists, whose photos and pictures (or their fragments) posted on the Internet, we used in the presentation.



Where we start: Evolving Big Data

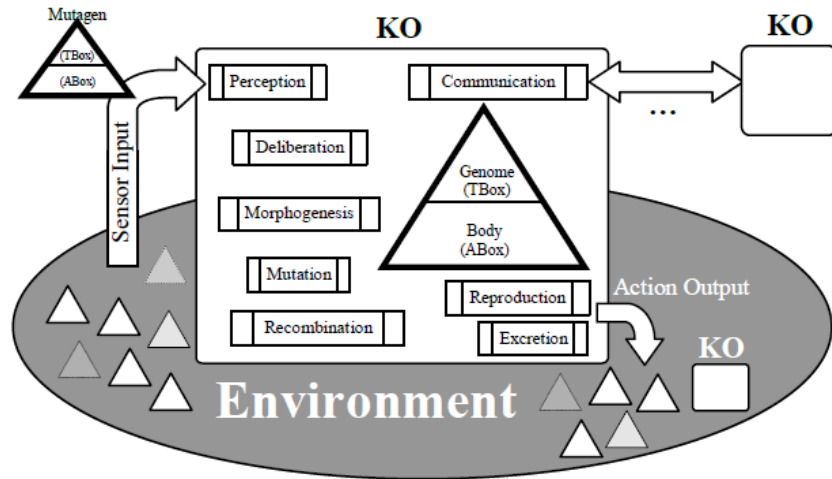
Ermolayev V., Akerkar R., Terziyan V., Cochez M., Towards Evolving Knowledge Ecosystems for Big Data Understanding, In: R. Akerkar (ed.), Big Data Computing, Chapman and Hall /CRC, 2014, 542 pp. (Chapter 1, pp. 3-55).

http://books.google.com.ua/books?hl=en&lr=&id=mXr6AQAAQBAJ&oi=fnd&pg=PA3&ots=E-GvQtCHlh&sig=iRH2WJZ_nebZ0kwaoDvmuioCq34&redir_esc=y#v=onepage&q&f=false

Among other objectives the paper aims at managing and searching within huge content of Big Data and Evolving Knowledge discovered out of it.

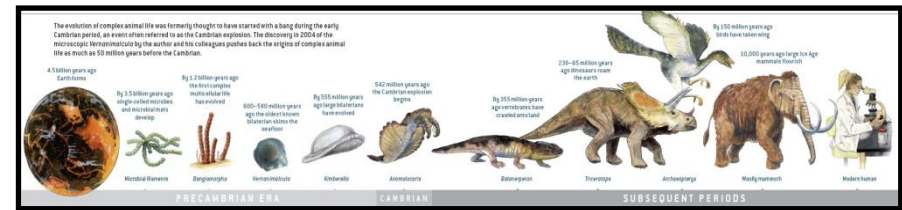


Big Data and Darwin Evolution



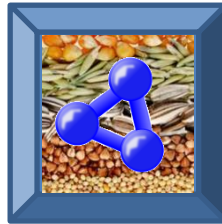
A Knowledge Organism (KO): functionality and environment. Small triangles of different transparency represent knowledge tokens in the environment – consumed and produced by KOs. These knowledge tokens may also be referred to as mutagens as they may trigger mutations

“The mechanisms of knowledge evolution are very similar to the mechanisms of biological evolution. Hence, the methods and mechanisms for the evolution of knowledge could be spotted from the ones enabling the evolution of living beings.”



Big Data is collected by different communities and its semantics is processed using naturally different ontologies. All these loosely coupled data and knowledge fractions in fact “live their own lives” based on very complex processes, i.e., evolve following the evolution of these cultures, their cognition mechanisms, standards, objectives, ontologies, etc. An infrastructure for managing and understanding such data straightforwardly needs to be regarded as an ecosystem of evolving processing entities. We propose treating ontologies (a key for understanding Big Data) as genomes and bodies of those knowledge processing entities. For that basic principles by Darwin are applied to their evolution aiming to get optimal or quasi-optimal populations of knowledge species. Those populations represent the evolving understanding of the respective islands of Big Data in their dynamics. This approach to knowledge evolution will require interpretation and implementation of concepts like “birth”, “death”, “morphogenesis”, “mutation”, “reproduction”, etc., applied to knowledge organisms, their groups, and environments.

Information Token vs. Knowledge Token



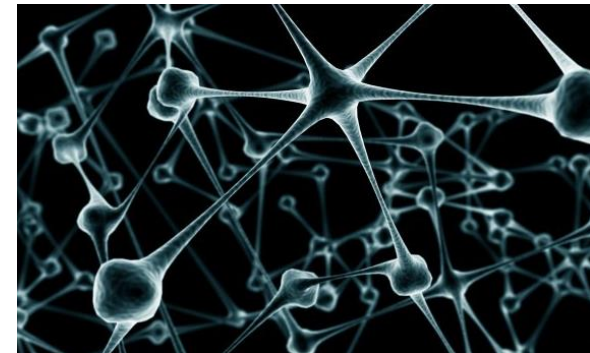
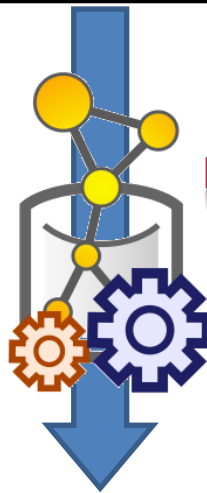
**INFORMATION
TOKEN**



The 1st KEYSTONE Open Conference (IKC-2015) will be held during 8-9 September 2015 at the University of Coimbra.

[Source: <http://www.keystone-cost.eu/>]

**KNOWLEDGE
EXTRACTION**



**KNOWLEDGE
TOKEN**

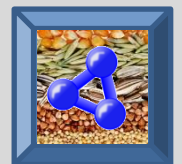


@prefix Keystone : <<http://www.keystone-cost.eu/Keystone.owl#>> .
Keystone: 1stKEYSTONEOpenConference

a

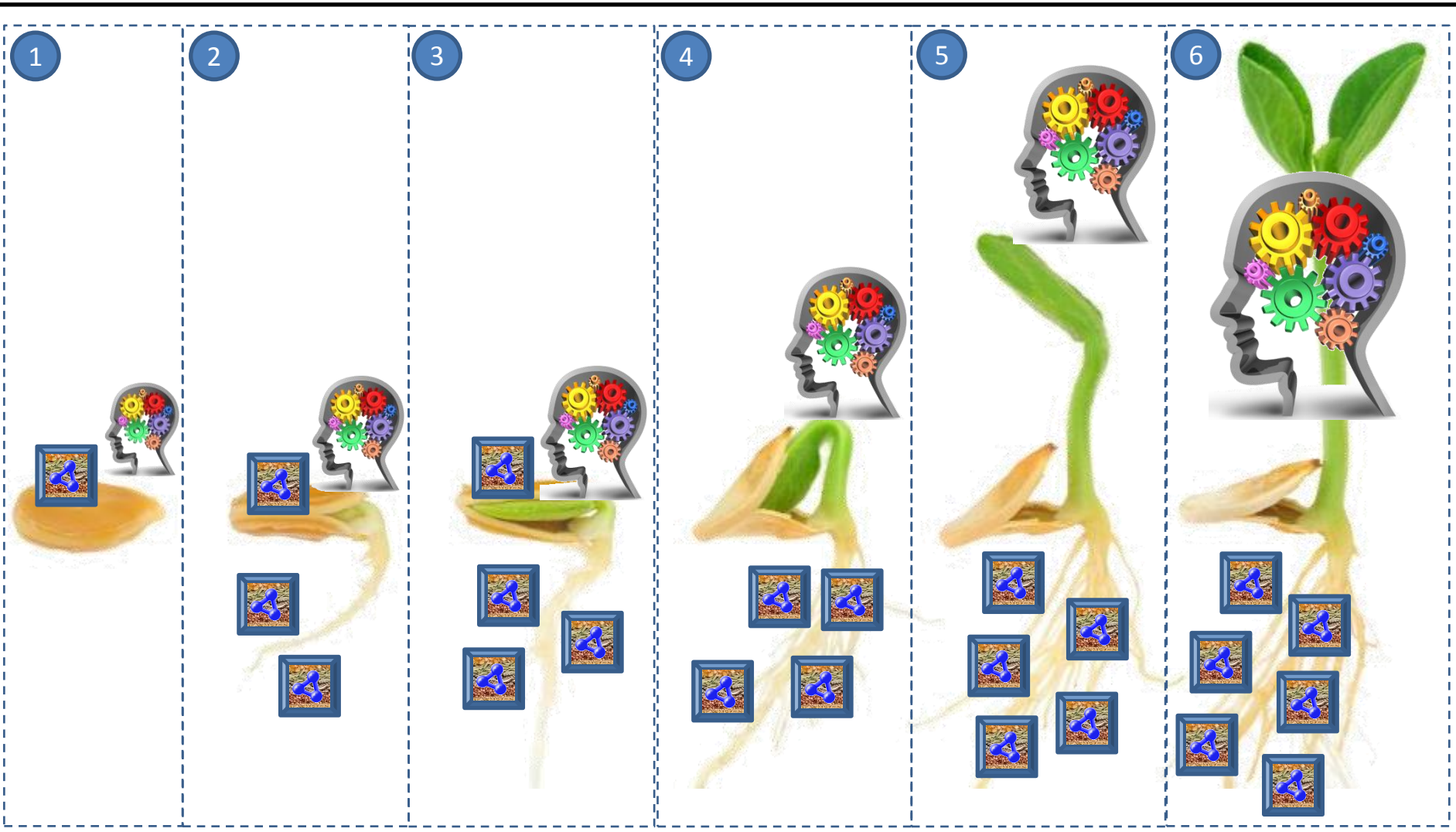
Keystone : hasAbbreviation
Keystone : hasStartingDate
Keystone : hasEndingDate
Keystone : hasHost

Keystone : Conference;
"IKC-2015" ;
"08/09/2015";
"09/09/2015";
Keystone : UniversityOfCoimbra .



n3

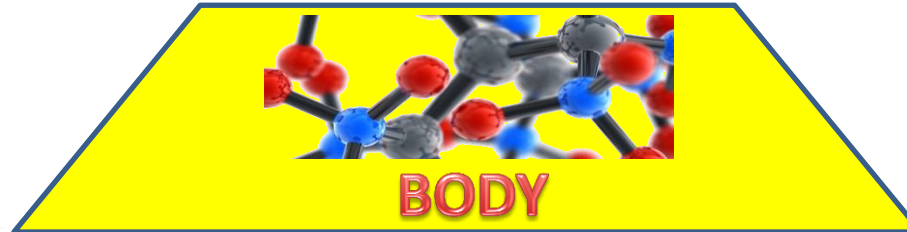
Knowledge Organism Consumes Knowledge Tokens and Evolves (grows incrementally)



Knowledge Organism



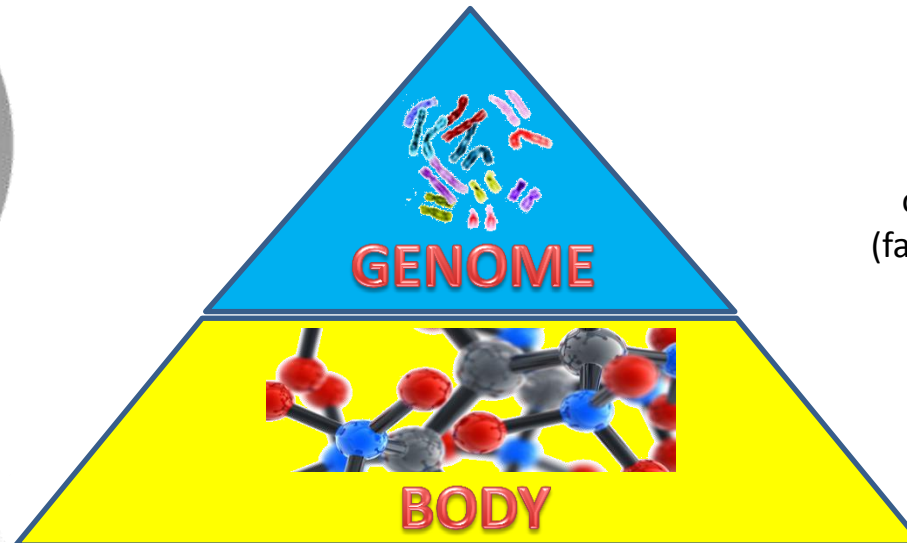
Knowledge Organism (has “Body”)



ABOX

Assertional Box
of the respective ontology
(facts about domain objects)

Knowledge Organism (has “Genome”)



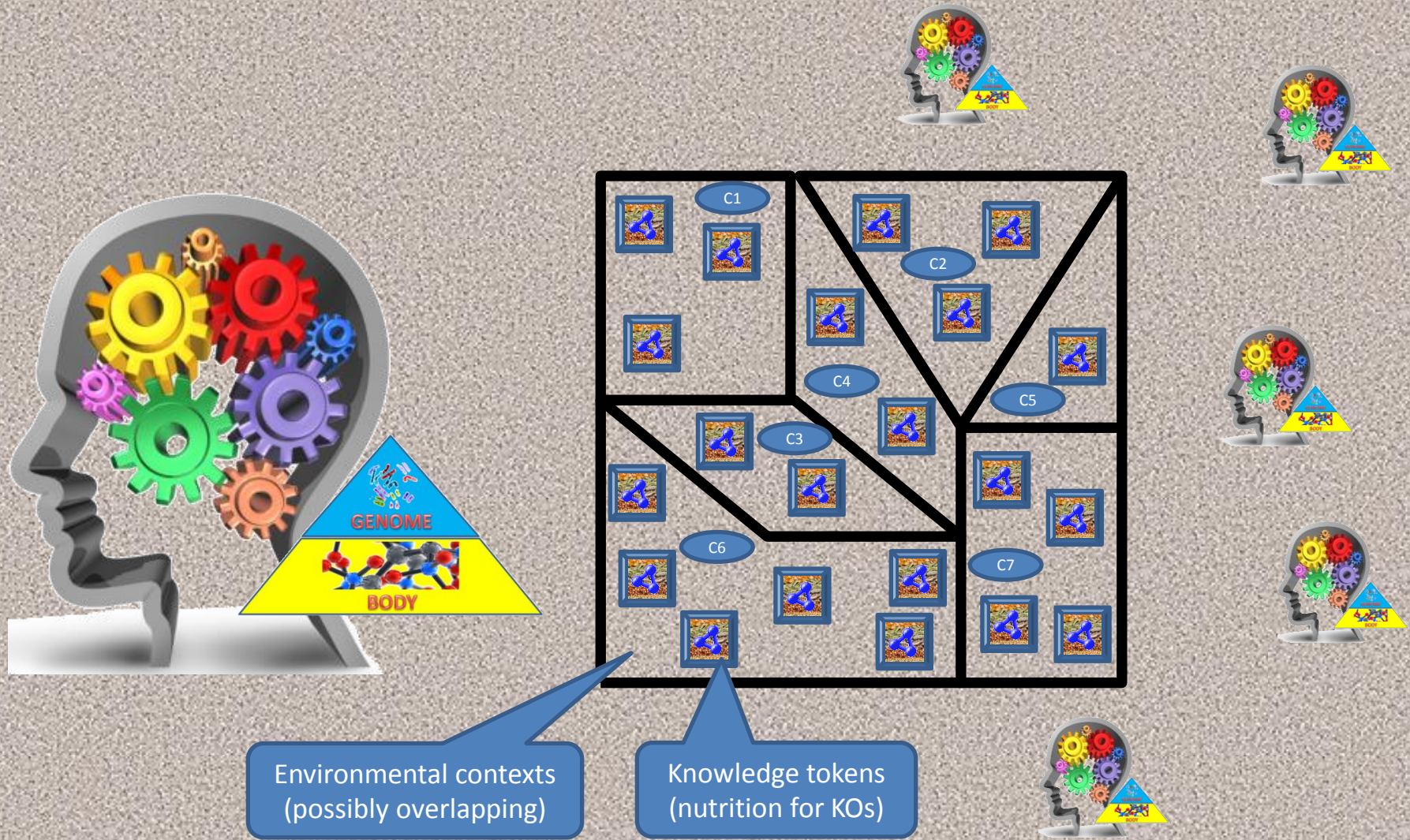
TBOX

Terminological Box
of the respective ontology
(facts about domain concepts)

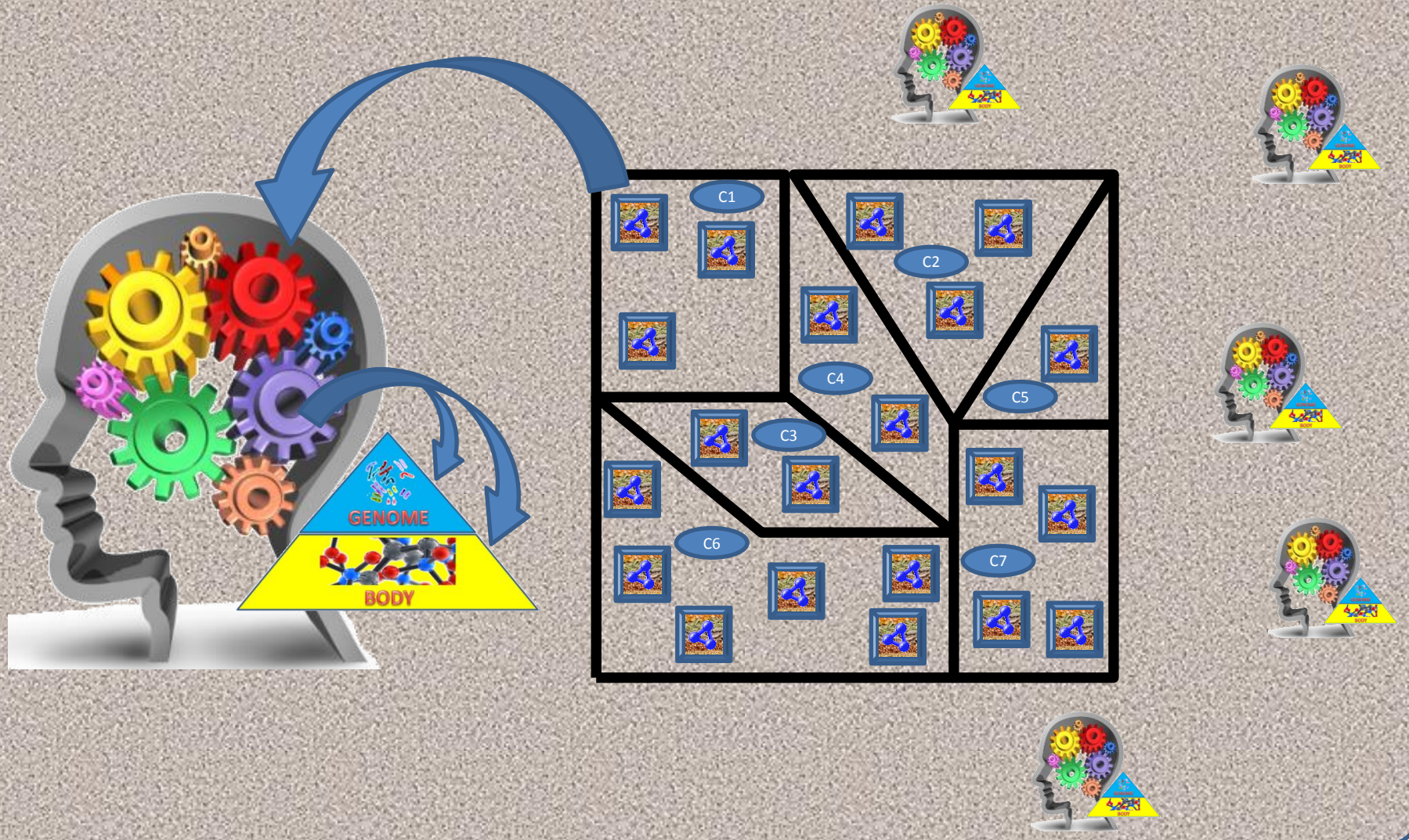
ABOX

Assertional Box
of the respective ontology
(facts about domain objects)

Knowledge Organism (Situated in the Environment)

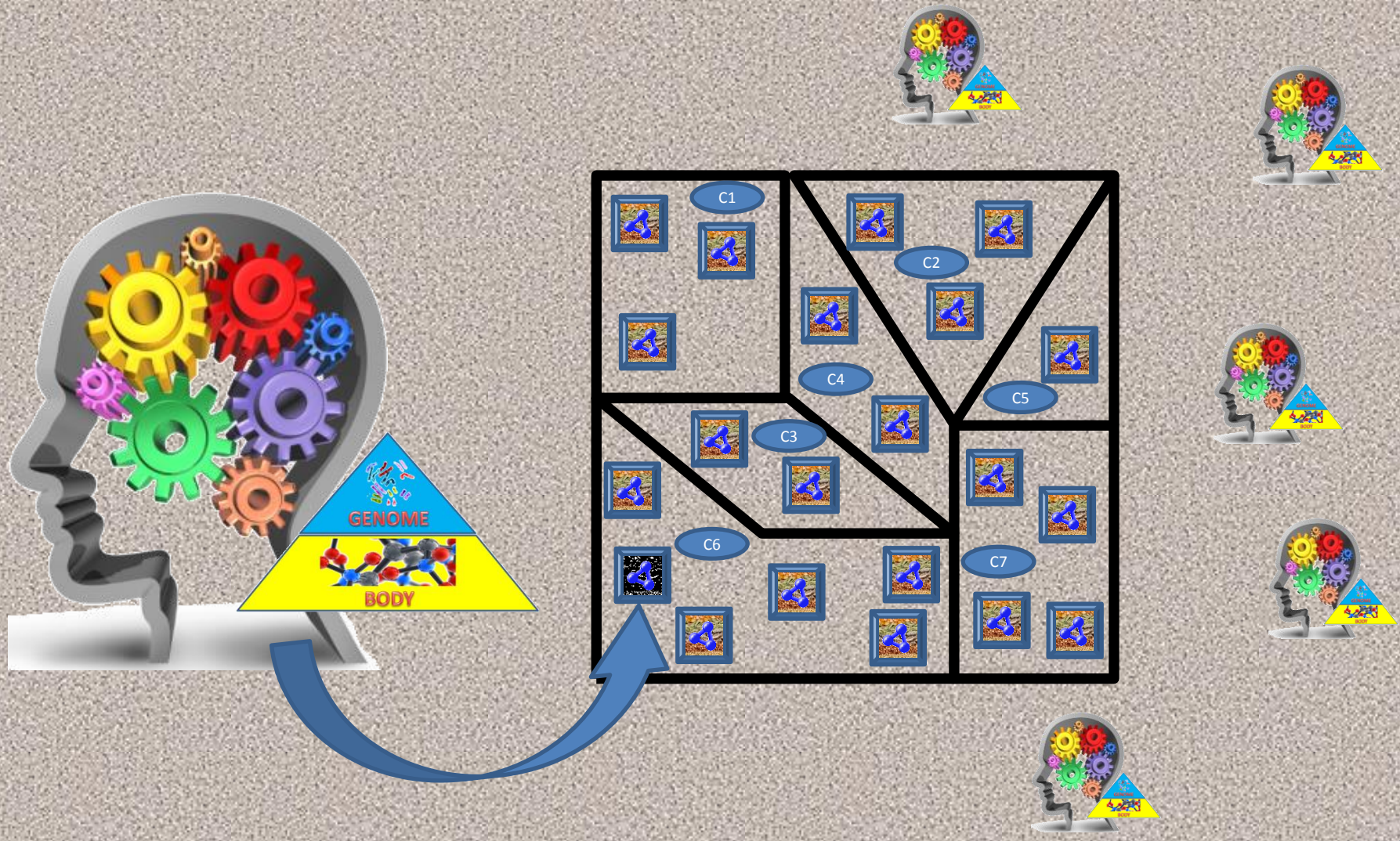


Knowledge Organism (consumes knowledge tokens ...



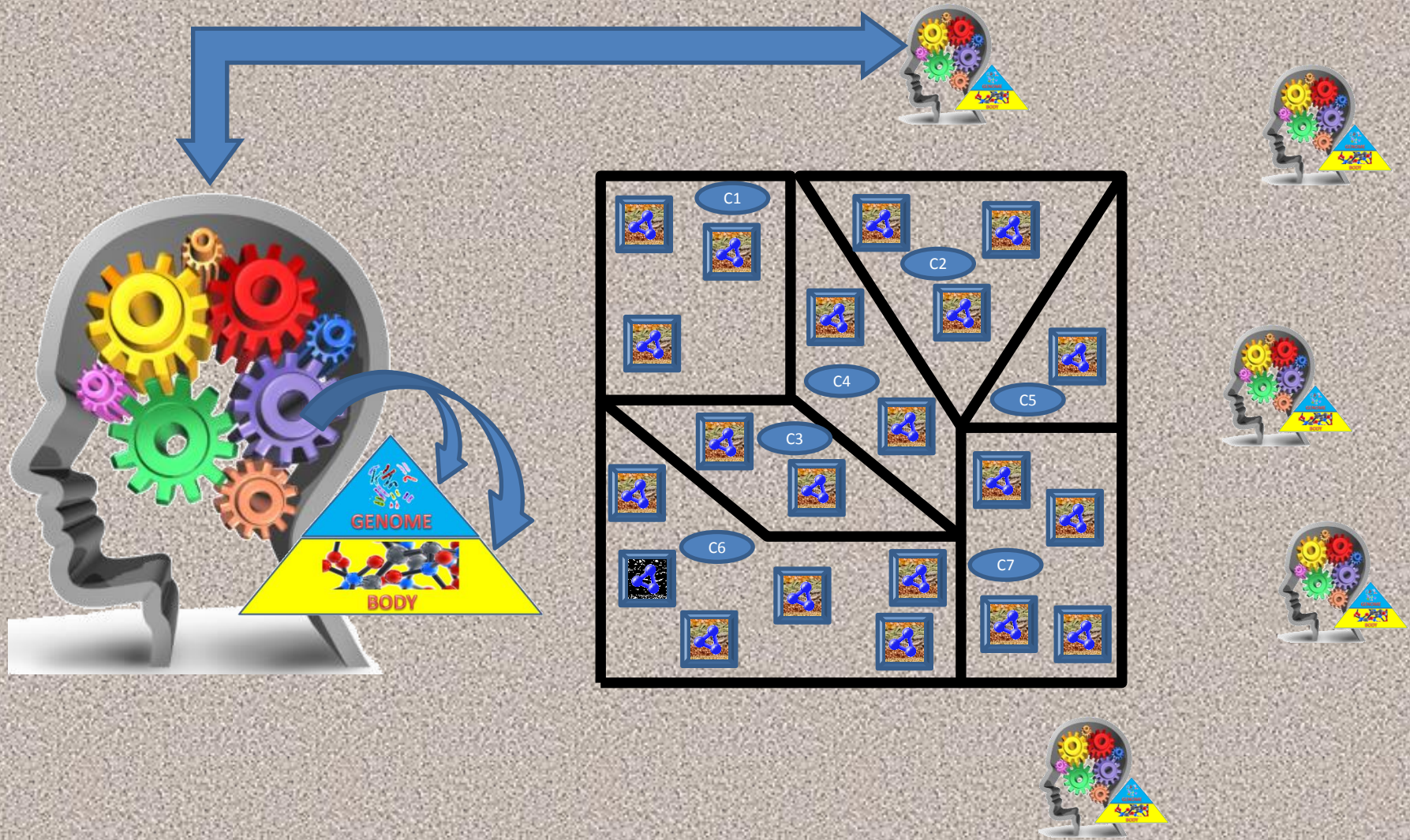
... and by doing that it modifies own genome and body)

Knowledge Organism (produces knowledge tokens ...



... and excretes it back to the environment)

Knowledge Organism (communicates with others ...



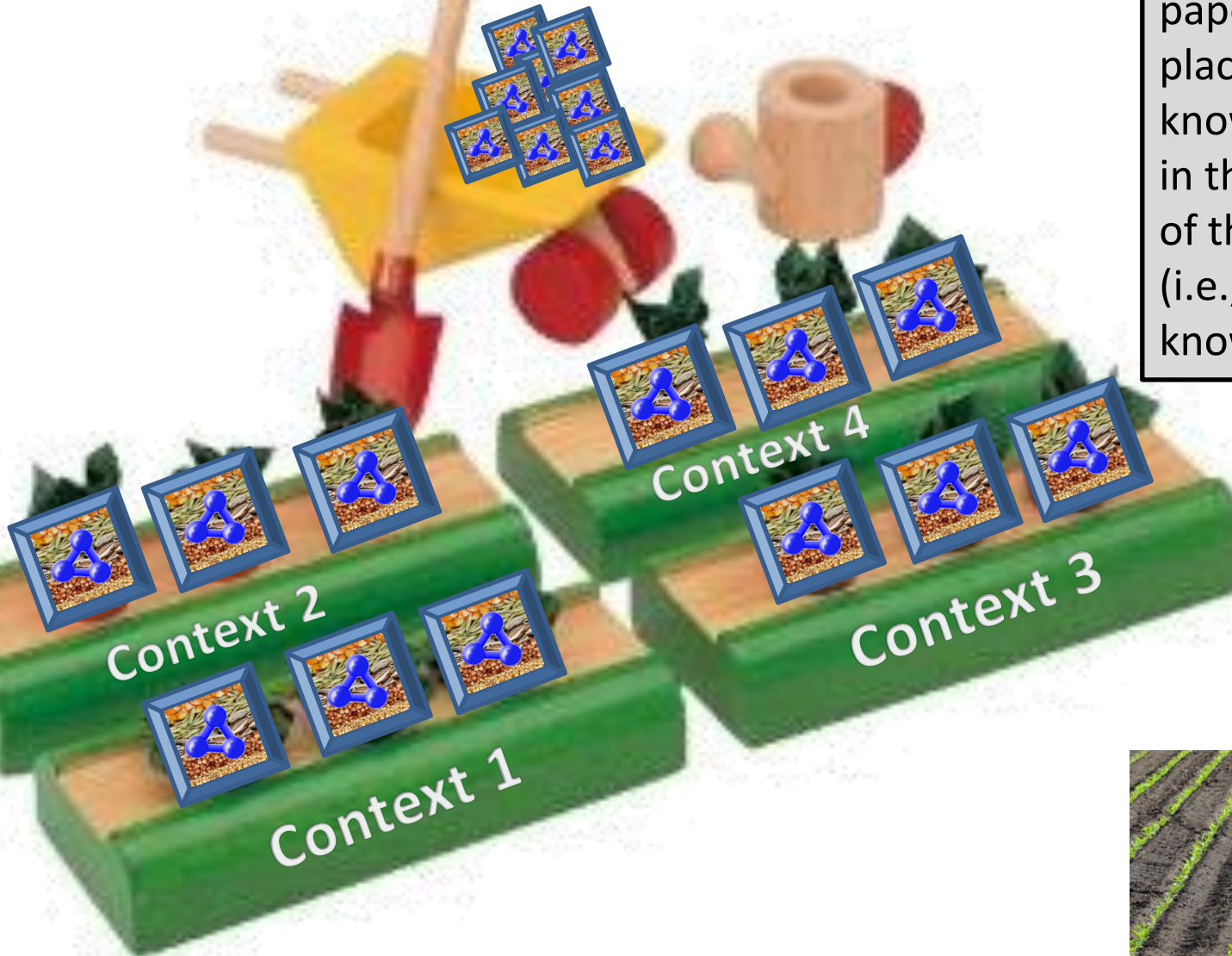
... and by doing that it modifies own genome and body)

“Sowing” Knowledge Tokens

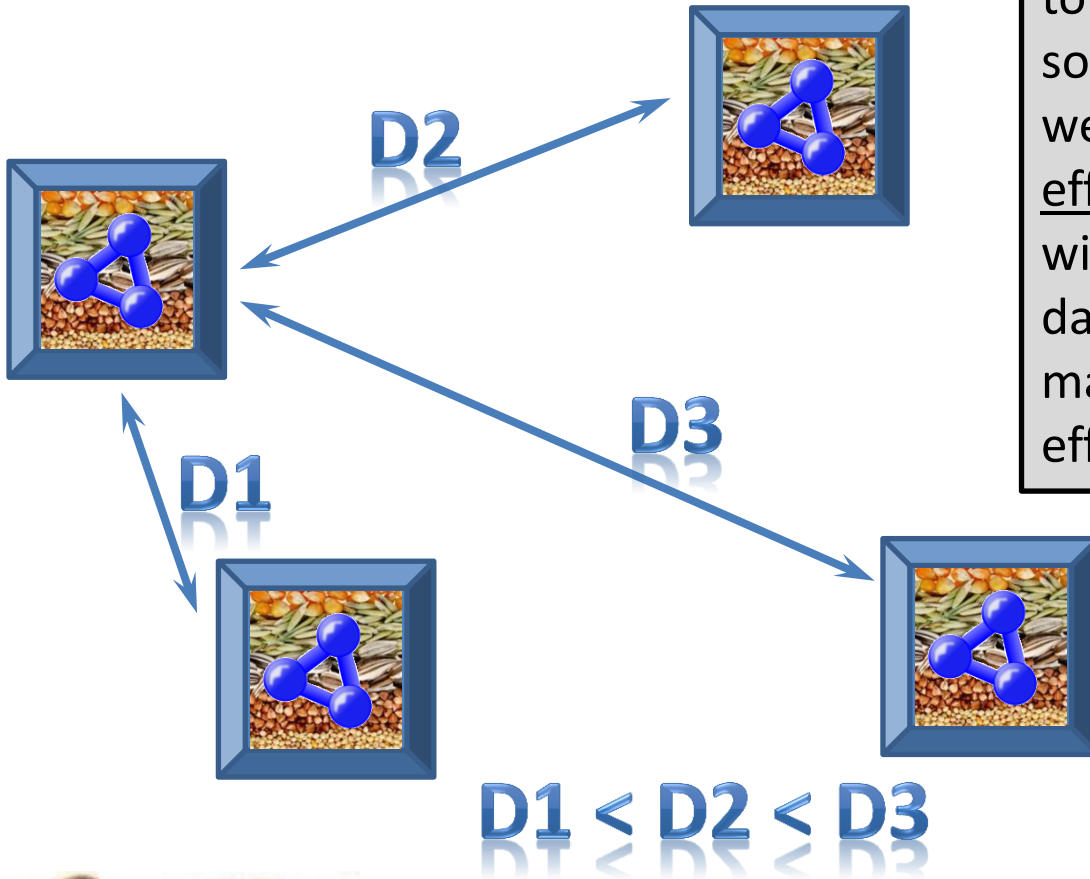


Sowing Knowledge Tokens Problem

“Sowing” in this paper stands for placing new knowledge tokens in the right contexts of the environment (i.e., contextualizing knowledge)



Sowing Knowledge Tokens Challenge



The main requirement for knowledge token sowing is that similar tokens get sown close to each other. Therefore we are dealing with the problem of an efficient similarity/distance search within large and high-dimensional data sets. Efficiency in this context may be interpreted as the ratio of the effort spent to the utility of the result.

We consider the problem of similarity search as one of finding pairs of tokens with relatively large intersection of the content.

Jaccard similarity: $J(A, B) = \frac{|A \cap B|}{|A \cup B|}$

There may be far too many pairs of items to test each pair for their degree of similarity, even if computing the similarity of any one pair can be done easily. That concern motivates a technique called “locality-sensitive hashing,” for focusing our search on pairs that are most likely to be similar.



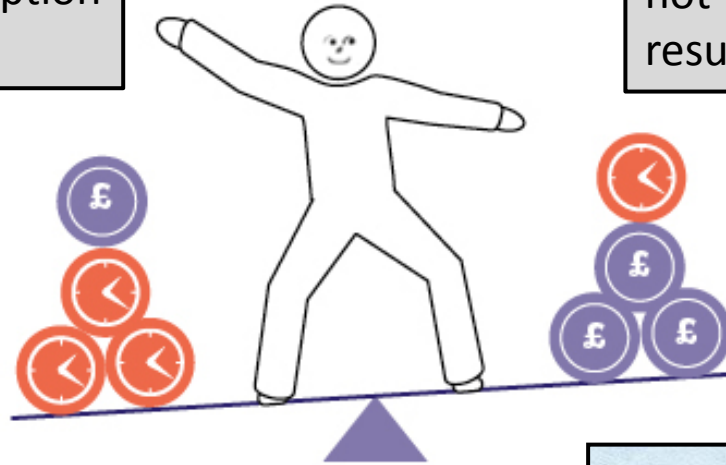
Efficiency vs. Effectiveness

(we are looking for a balanced solution)

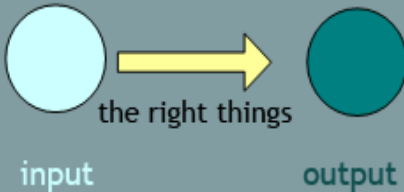
Effectiveness is achieved if:

- (a) not a single important knowledge token is left unattended (completeness); and
- (b) these tokens are sowed adequately for further consumption (expressiveness/granularity).

Efficiency is achieved if the ratio of the effort (resource) spent is reasonable comparably to the utility of the result. E.g., if a result is not timely the utility of the resulting knowledge will drop.



Effective



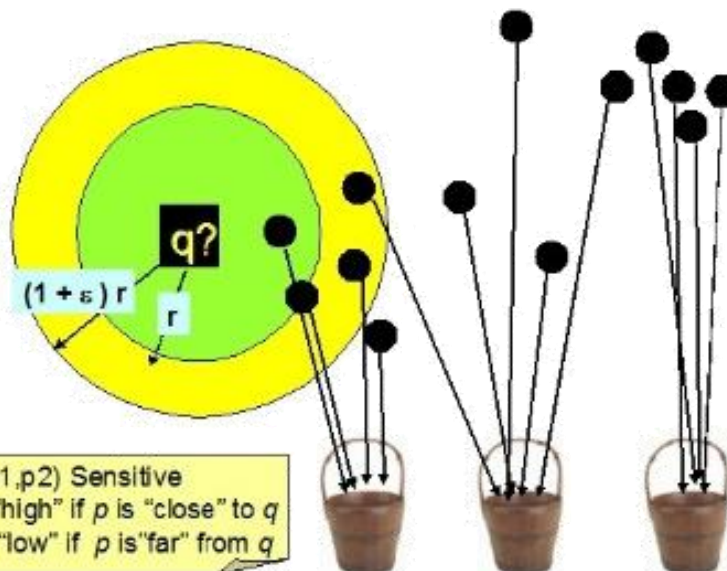
Efficient



Locality-Sensitive Hashing

- LSH is an efficient approximate nearest neighbor search method in a large data set. Instead of comparing all pairs of items within a set, items are hashed into buckets (multiple times), such that similar items will be more likely to hash into the same buckets. As a result, the number of comparisons needed will be reduced; only the items within any one bucket will be compared.

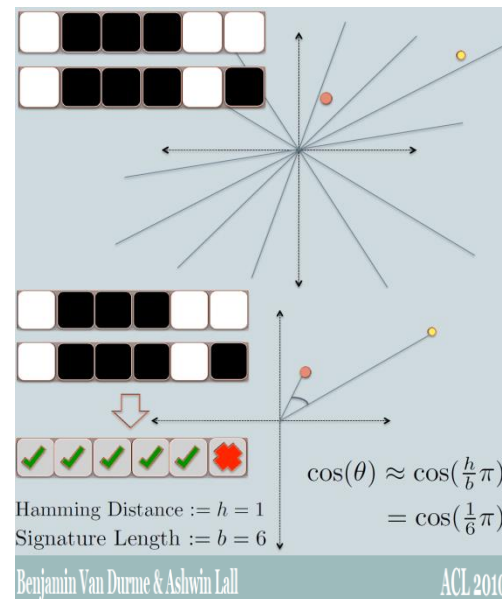
LSH - a collection of techniques for using hash functions to map large sets of objects down to smaller hash values in such a way that, when two objects have a small distance from each other, their hash values are likely to be the same.



(r, ϵ, p_1, p_2) Sensitive
 $P_1 = \Pr[l(p)=l(q)]$ is "high" if p is "close" to q
 $P_2 = \Pr[l(p)=l(q)]$ is "low" if p is "far" from q



$$J(A, B) = \frac{|A \cap B|}{|A \cup B|}$$



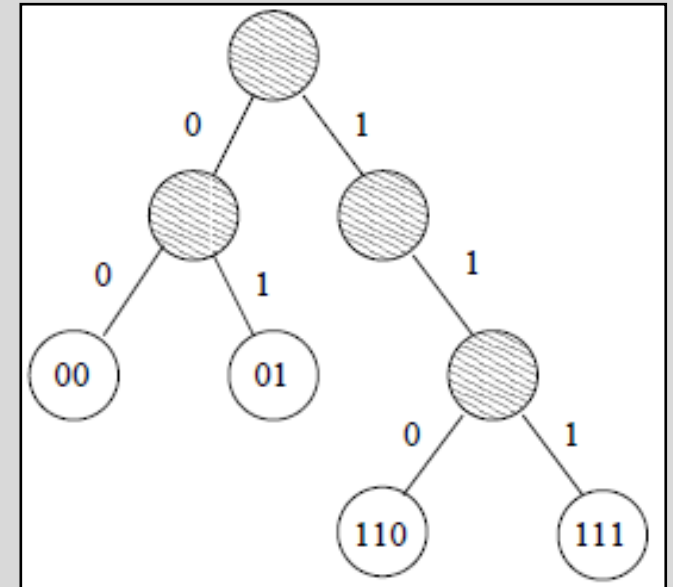
LSH Forest

LSH Forests were introduced in [A] and its essential improvement over LSH is that in LSH Forest the points do not get a fixed-length labels. Instead, the length of the label is decided for each point individually.

[A] Bawa, M., Condie, T., Ganesan, P.: LSH forest: self-tuning indexes for similarity search. In: Proceedings of the 14th international conference on World Wide Web, ACM (2005) 651- 660.

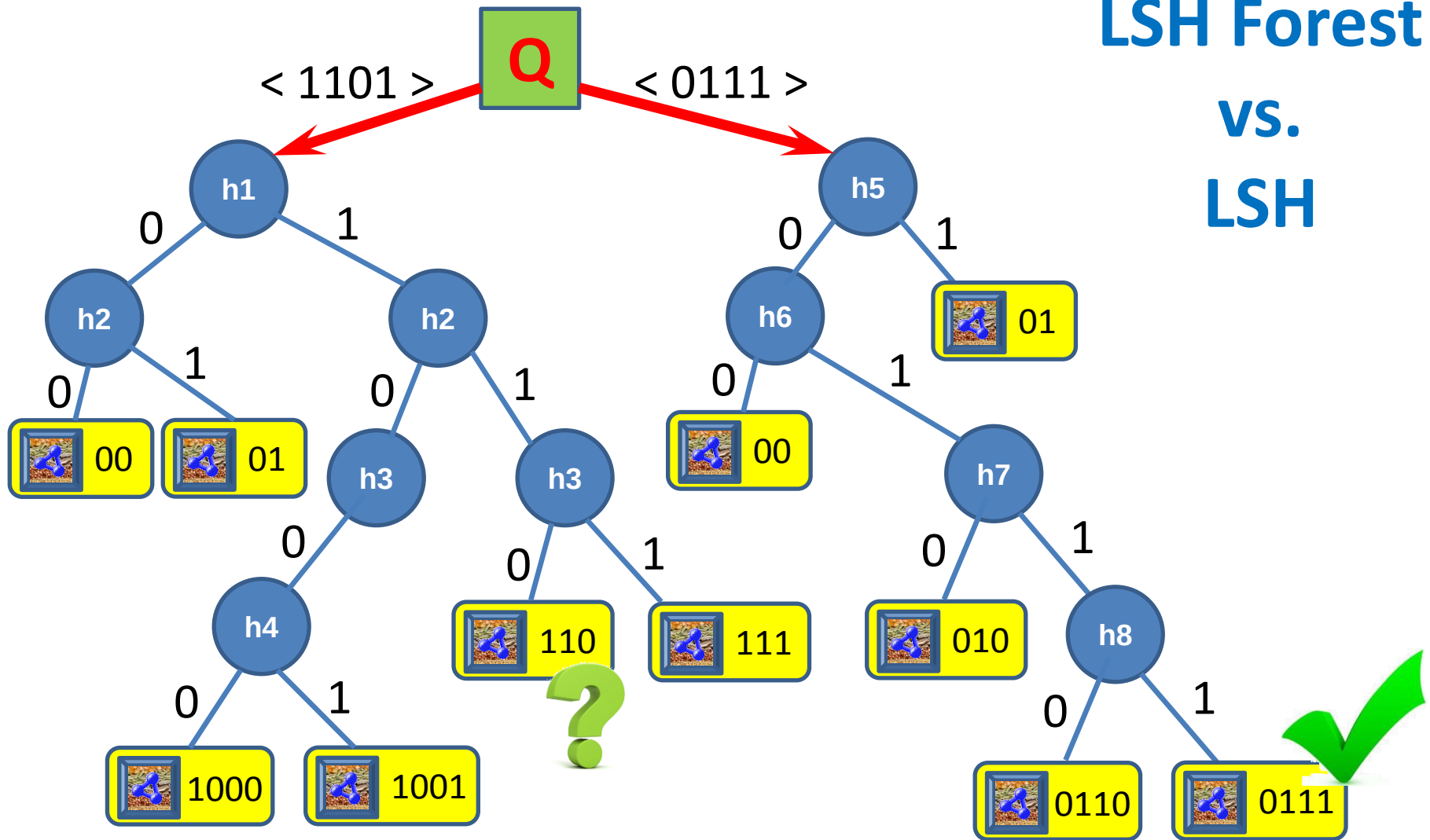
The “best” labels (indexes) would have the following properties:

- ✓ A. *Accuracy*: The set of candidates retrieved by the index should contain the most similar objects to the query.
- ✓ B. *Efficient Queries*: The number of candidates retrieved must be as small as possible, to reduce I/O and computation costs.
- ✓ C. *Efficient Maintenance*: The index should be built in a single scan of the dataset, and subsequent inserts and deletes of objects should be efficient.
- ✓ D. *Domain Independence*: The index should require no effort on the part of an administrator to get it working on any data domain; there should be no special tuning of parameters required for each specific dataset.
- ✓ E. *Minimum Storage*: The index should use as little storage as possible, ideally linear in the data size.



The Figure shows an example of LSH Tree that contains four points, with each hash function producing one bit as output. The leafs of the tree correspond to the four points, with their labels marked inside. The shaded circles correspond to internal nodes. Observe that the label of each leaf simply represents the path to it from the root. Also observe that not all internal nodes need to have two children; some internal nodes may have only one child (for example, the right child of the root). In general, there is no limit on the number of internal nodes in a prefix tree with n leaves, since we can have long chains of internal nodes.

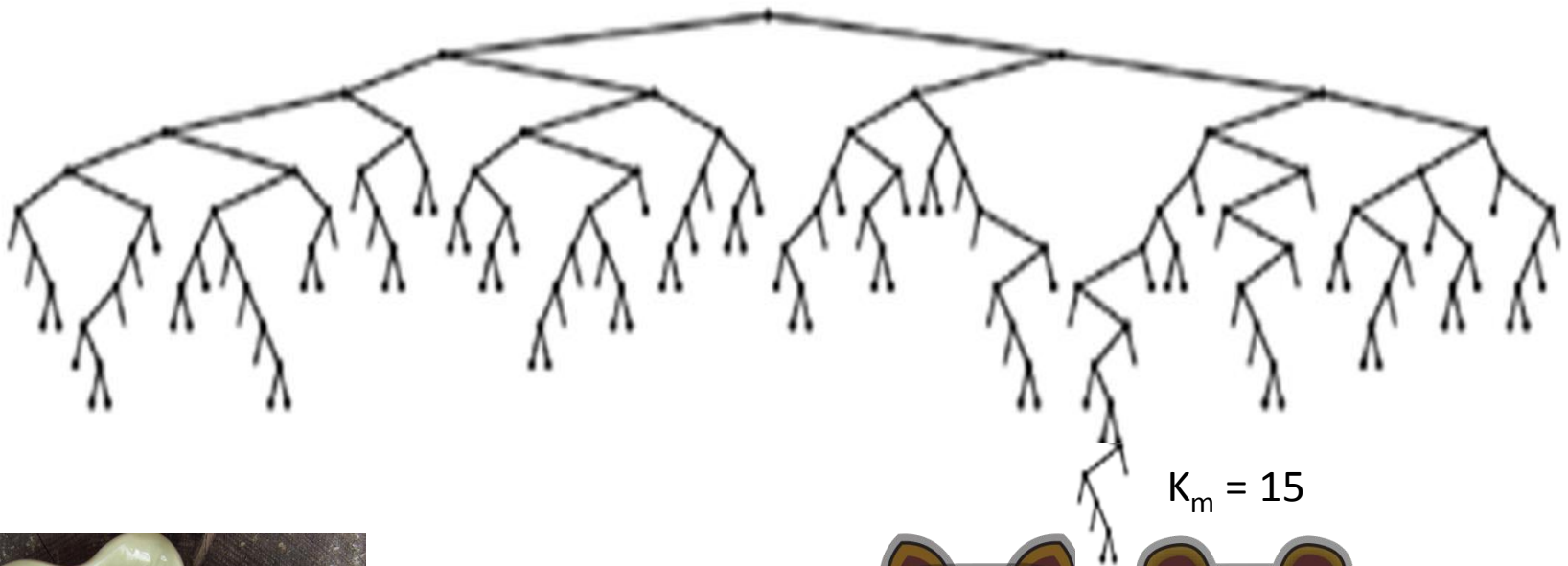
LSH Forest vs. LSH



LSH forest represents each hash table, built from LSH, using a tree, by pruning subtrees (nodes) that do not contain any database points and also restricting the depth of each leaf node not larger than a threshold. Different from the conventional scheme that finds the candidates from the hash buckets corresponding to the hash codes of the query point, the search algorithm finds the points contained in subtrees over LSH forest having the largest prefix match with the hash code of the query.

The challenge of very close points in LSH Forest

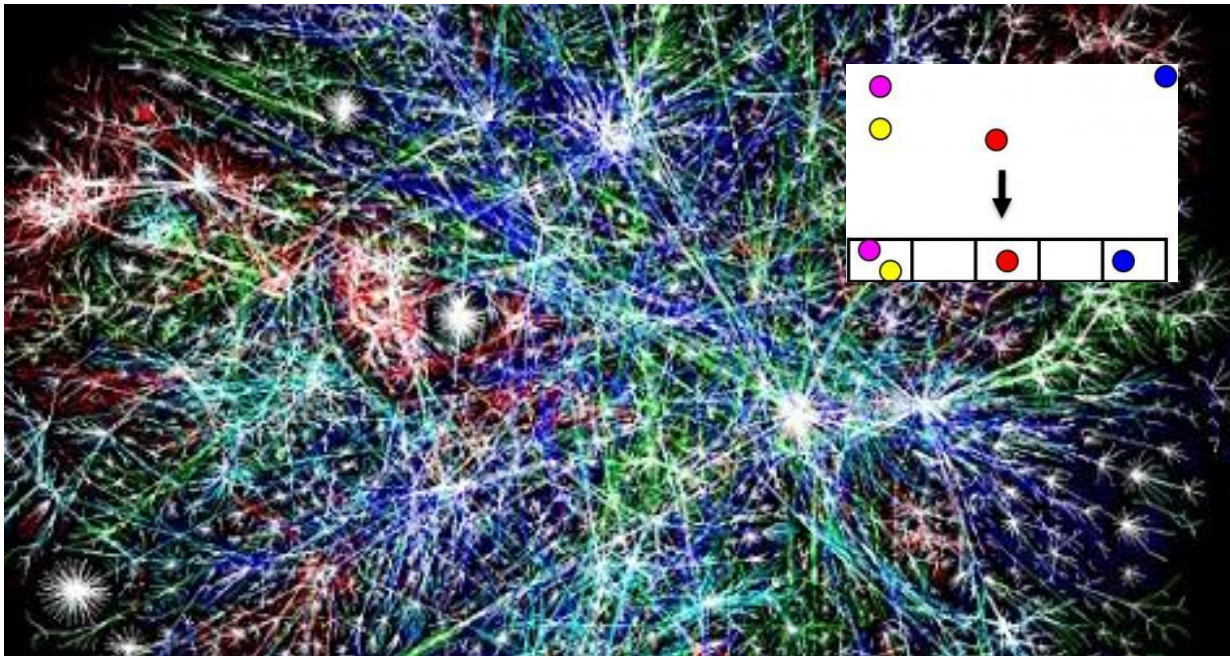
One issue with both the conventional LSH and LSH Forest is that, using hashing, it is not possible to distinguish between arbitrary close points. Therefore, LSH assumes a minimum distance between any two points and LSH Forest defines a maximum label length. This maximum label length is equal to the maximum height of the tree and is indicated as k_m .



We have checked the LSH with the Big Ontologies already before; now its time to check the Forest

Cochez, M. (2014). **Locality-sensitive hashing for massive string-based ontology matching**. In D. Ślęzak, B. Dunin-Kępicz, M. Lewis, & T. Terano (Eds.), 2014 IEEE/WIC/ACM International Joint Conferences on Web Intelligence (WI) and Intelligent Agent Technologies (IAT) (pp. 134-140). IEEE. doi:10.1109/WI-IAT.2014.26 http://users.jyu.fi/~miselico/papers/LSH_ontology_matching.pdf

Among other objectives the paper above aims at the use of locality-sensitive hashing (LSH) for string-based matching of big ontologies





Ontology Alignment Evaluation Initiative

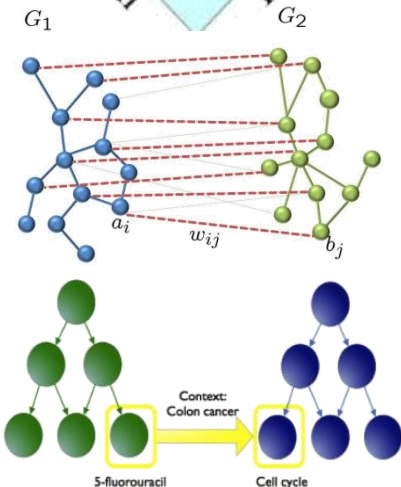
<http://www.cs.ox.ac.uk/isg/projects/SEALS/oeai/2013/>

The OAEI organizes evaluation campaigns aiming at **evaluating ontology matching** technologies. The OAEI campaign confronts ontology matchers to ontology and data sources to be matched. It provides test sets for testing and evaluation and offers the possibility to compare different matching tools.

The **Large Biomedical Ontologies** track of the OAEI consists of finding alignments between the Foundational Model of Anatomy (**FMA**), **SNOMED** CT, and the National Cancer Institute Thesaurus (**NCI**). These ontologies are semantically rich and contain tens of thousands of classes.

UMLS Metathesaurus has been selected as the basis for the track reference alignments. UMLS is currently the most comprehensive effort for integrating independently-developed medical thesauri and ontologies, including FMA, SNOMED CT, and NCI. The current (2015AA) Metathesaurus contains more than 3.1 million concepts and 12 million unique concept names from over 170 source vocabularies.

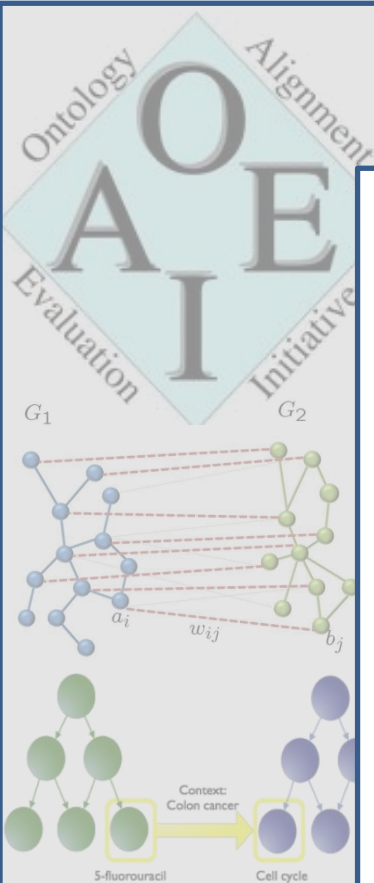
The UMLS-based reference alignments are used as a gold standard (e.g., for computing Precision and Recall in our experiments). From these reference alignments, in our paper, we only retain the equal correspondences, and all the confidence levels are set to one. This choice of datasets is made mainly to enable the comparison of matching systems. Further reasons include the availability of reference alignments and the fact that these ontologies are reasonably large.



Ontology Alignment Evaluation Initiative

<http://www.cs.ox.ac.uk/isg/projects/SEALS/oeai/2013/>

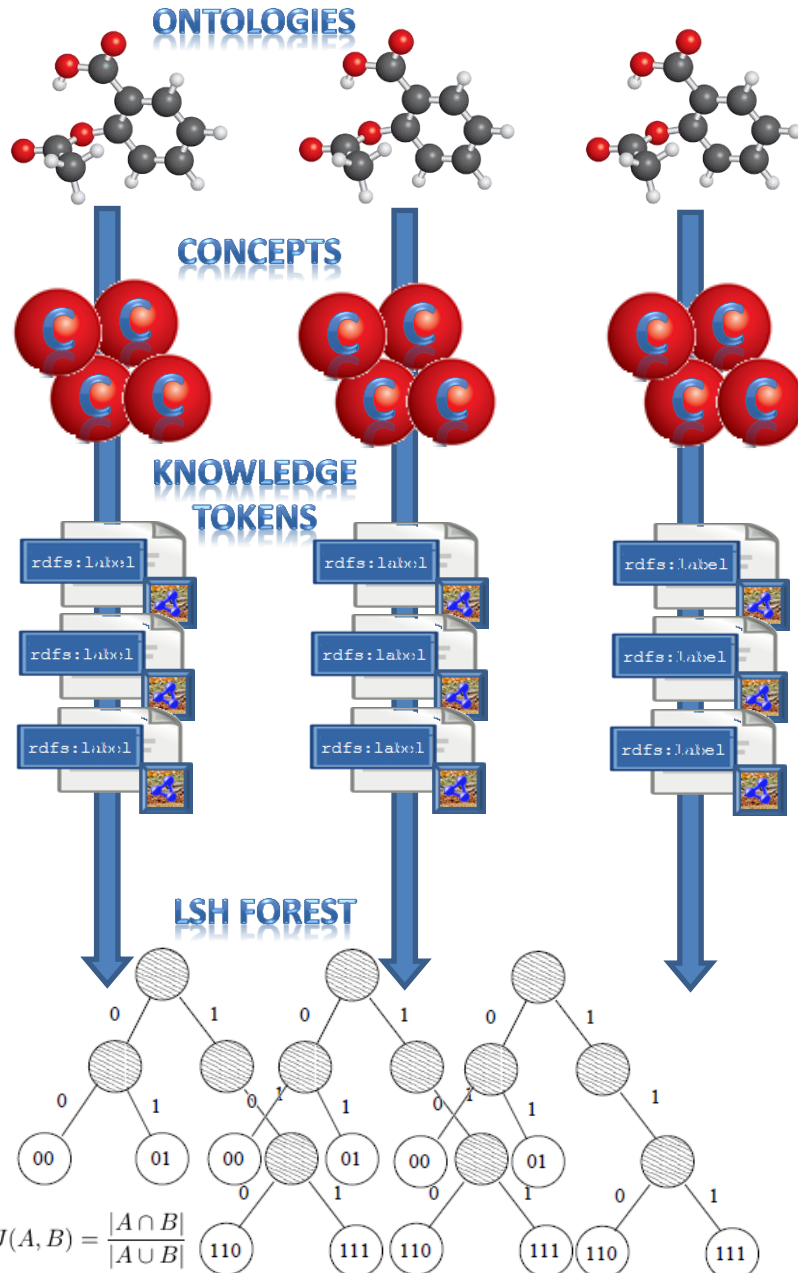
THE FOCUS OF THIS PAPER IS **NOT ONTOLOGY ALIGNMENT PROBLEM, BUT WE ARE USING LARGE ONTOLOGIES FROM OAEI AS A SOURCE OF KNOWLEDGE TOKENS FOR OUR EXPERIMENTS AND AS A “GOLDEN STANDARD” FOR THE PRECISION AND RECALL CALCULATIONS WHEN COMPARING EFFICIENCY AND EFFECTIVENESS OF LSH VS. LSH-FOREST**



vocabularies and standards to enable interoperability between computer systems



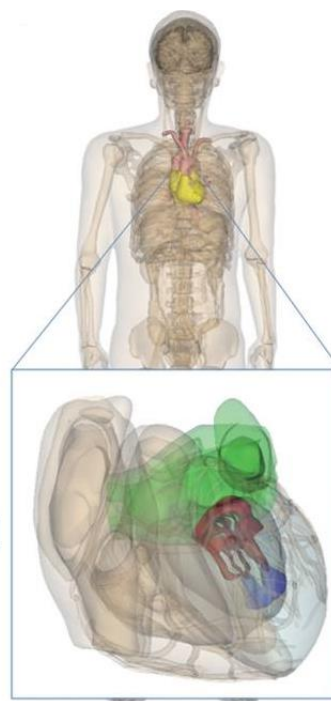
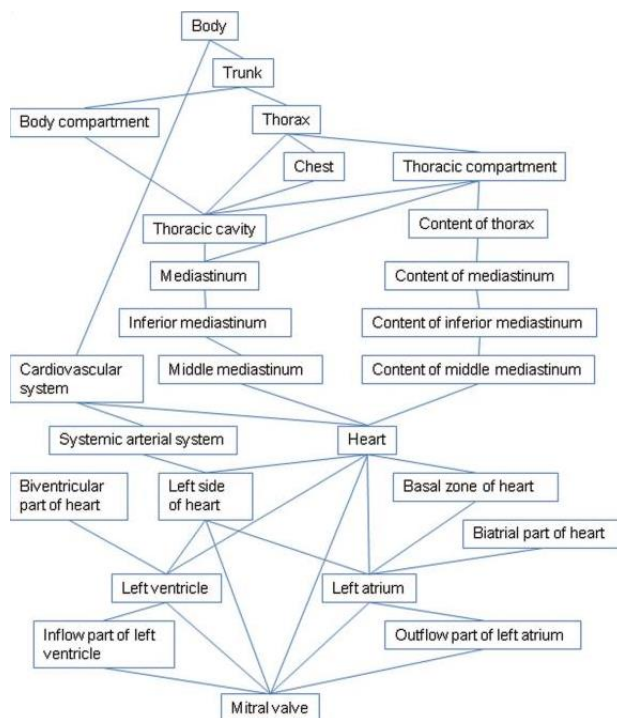
Experiments (basic settings)



The experiments are designed so that we start from a simple set-up and more complexity is added in each following experiment. In the first series of experiments, we feed class representations of three different ontologies into an LSH tree and present several metrics of how they are spread over the tree. We use two and later three ontologies and measure how the LSH Forest classifies the concepts and how it is capable of aligning the ontologies. Finally, in the third series we add dynamism by sampling the ontologies in different ways and measure how the memory usage and processing time evolve. In the experiments labeled classes are used as an input to the LSH Forest data structure. Classes with similar labels are placed closer together in the trees. A knowledge token (in the experiments) is represented with the text of the **rdfs:label**, there is a distance function (**Jaccard distance**), there are hash functions which are likely to give the same outcome for close things (=similar texts). Finally, things with similar hash outcomes are placed together.

The FMA Ontology

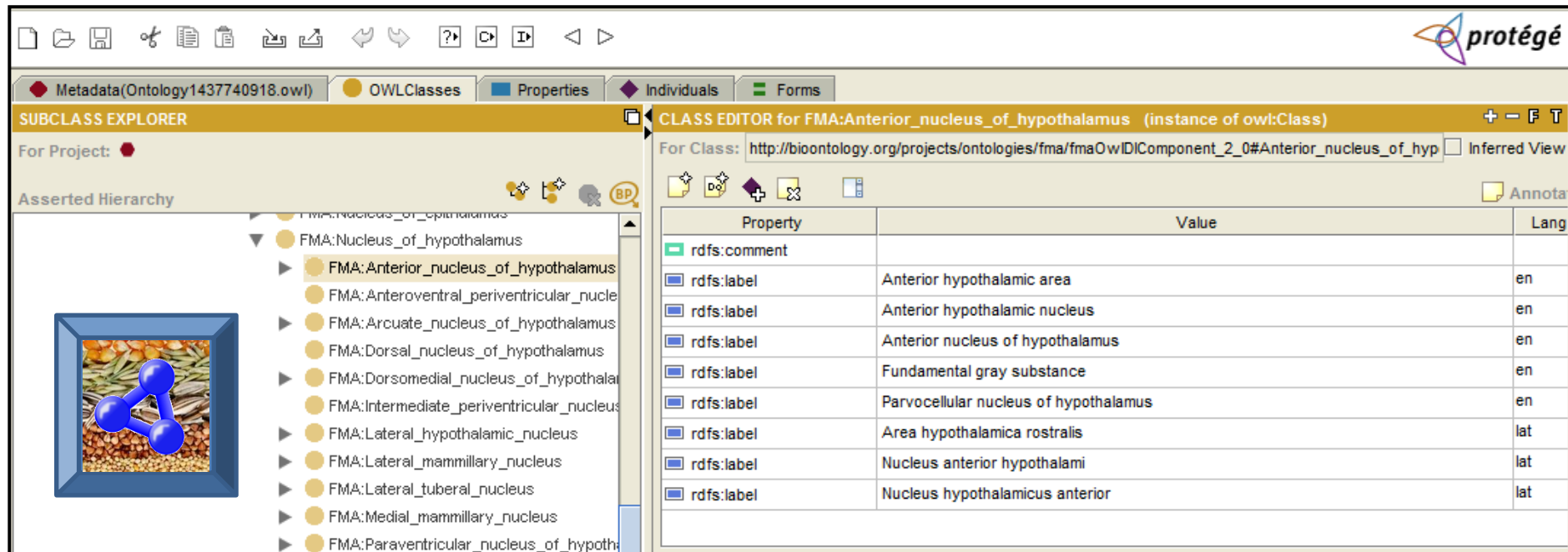
Foundational Model of Anatomy



The FMA ontology contains 78,989 classes and non-hierarchical datatype properties, i.e., neither object or datatype properties nor instances.

The Foundational Model of Anatomy Ontology (**FMA**) is an evolving computer-based knowledge source for biomedical informatics; it is concerned with the representation of classes or types and relationships necessary for the symbolic representation of the phenotypic structure of the human body in a form that is understandable to humans and is also navigable, parseable and interpretable by machine-based systems. Specifically, the FMA is a domain ontology that represents a coherent body of explicit declarative knowledge about human anatomy. FMA contains also the official source for anatomical terms with the intent to standardize them. The FMA represents concepts (classes) and associates with them a unique identification number and also all the terms that we can find in anatomical discourse and the literature referring to this class. The preferred name serves as the most useful identifier of a class, because (unlike a numerical ID) it is meaningful to humans and is also processable by machines (computers) without ambiguity. The latter property requires that both preferred names and synonyms uniquely refer only to one class.

The FMA Ontology



Subclass Explorer

For Project:

Asserted Hierarchy

- FMA:Nucleus_of_hypothalamus
 - FMA:Anterior_nucleus_of_hypothalamus
 - FMA:Anteroventral_periventricular_nucleus
 - FMA:Arcuate_nucleus_of_hypothalamus
 - FMA:Dorsal_nucleus_of_hypothalamus
 - FMA:Dorsomedial_nucleus_of_hypothalamus
 - FMA:Intermediate_periventricular_nucleus
 - FMA:Lateral_hypothalamic_nucleus
 - FMA:Lateral_mammillary_nucleus
 - FMA:Lateral_tuberal_nucleus
 - FMA:Medial_mammillary_nucleus
 - FMA:Paraventricular_nucleus_of_hypothalamus

Class Editor for FMA:Anterior_nucleus_of_hypothalamus (instance of owl:Class)

For Class: http://bioontology.org/projects/ontologies/fma/fmaOwlIDComponent_2_0#Anterior_nucleus_of_hyp ☐ Inferred View

Property	Value	Lang
rdfs:comment		
rdfs:label	Anterior hypothalamic area	en
rdfs:label	Anterior hypothalamic nucleus	en
rdfs:label	Anterior nucleus of hypothalamus	en
rdfs:label	Fundamental gray substance	en
rdfs:label	Parvocellular nucleus of hypothalamus	en
rdfs:label	Area hypothalamica rostralis	lat
rdfs:label	Nucleus anterior hypothalami	lat
rdfs:label	Nucleus hypothalamicus anterior	lat

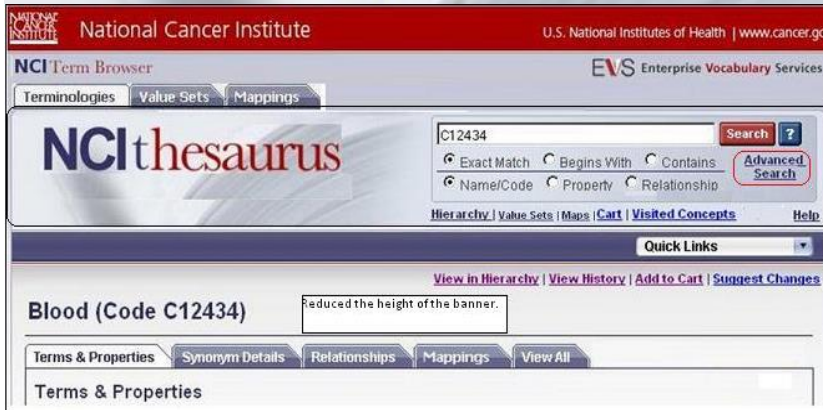
```
<owl:Class rdf:about="http://bioontology.org/projects/ontologies/fma/fmaOwlIDComponent_2_0#Anterior_nucleus_of_hypothalamus">
  <rdfs:label xml:lang="en">Anterior hypothalamic area</rdfs:label>
  <rdfs:label xml:lang="en">Anterior hypothalamic nucleus</rdfs:label>
  <rdfs:label xml:lang="en">Anterior nucleus of hypothalamus</rdfs:label>
  <rdfs:label xml:lang="en">Fundamental gray substance</rdfs:label>
  <rdfs:label xml:lang="en">Parvocellular nucleus of hypothalamus</rdfs:label>
  <rdfs:label xml:lang="lat">Area hypothalamica rostralis</rdfs:label>
  <rdfs:label xml:lang="lat">Nucleus anterior hypothalami</rdfs:label>
  <rdfs:label xml:lang="lat">Nucleus hypothalamicus anterior</rdfs:label>
  <rdfs:subClassOf rdf:resource="http://bioontology.org/projects/ontologies/fma/fmaOwlIDComponent_2_0#Nucleus_of_hypothalamus"/>
</owl:Class>
```

The NCI Ontology

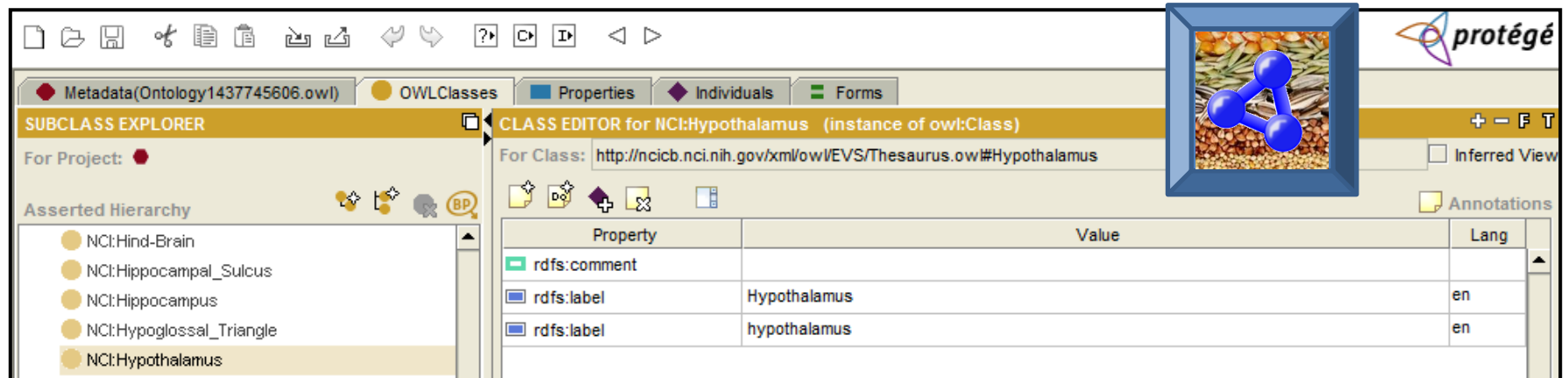


NCI thesaurus

Version: 15.06e (Release date: 2015-06-29)



The NCI Thesaurus is an ontology-like vocabulary that includes broad coverage of the cancer domain, including cancer related diseases, findings and abnormalities; anatomy; agents, drugs and chemicals; genes and gene products and so on. In certain areas, like cancer diseases and combination chemotherapies, it provides the most granular and consistent terminology available. It combines terminology from numerous cancer research related domains, and provides a way to integrate or link these kinds of information together through semantic relationships. The NCI ontology currently contains 66,724 concepts (classes) structured into 20 taxonomic trees. The NCI ontology contains classes, non-hierarchical datatype and hierarchical object properties.



The SNOMED Ontology

<http://snomedinaction.org/>

SNOMED CT

The global
language of
healthcare

IHTSDO
Leading healthcare
terminology, worldwide



Welcome to IHTSDO

<http://www.ihtsdo.org/>

The International Health Terminology Standards Development Organisation determines global standards for health terms, an essential part of improving the health of humankind.

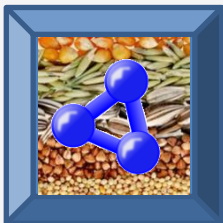
We are committed to maintain and grow our leadership as the global experts in healthcare terminology, ensuring that **SNOMED CT**, our world-leading product, is accepted as the global common language for health terms.

SNOMED CT is the most comprehensive and precise clinical health terminology product in the world, owned and distributed around the world by The International Health Terminology Standards Development Organization (IHTSDO). SNOMED CT is a systematically organized computer processable collection of medical terms providing codes, terms, synonyms and definitions used in clinical documentation and reporting. It provides a standardized way to represent clinical phrases captured by the clinician and enables automatic interpretation of these. SNOMED CT is a clinically validated, semantically rich, controlled vocabulary that facilitates evolutionary growth in expressivity. We used fragment, which contains 122,464 concepts (classes) from the SNOMED ontology.

Parent(s):
(Select a parent to make it the "Current Concept".)
[Viral upper respiratory tract infection \(disorder\)](#)

Current Concept:
[Common cold \(disorder\)](#)

Child(ren):
(N=0) (Select a child to make it the "Current Concept".)



Current Concept:

Fully Specified Name: Common cold (disorder)
ConceptId: 82272006

Defining Relationships:

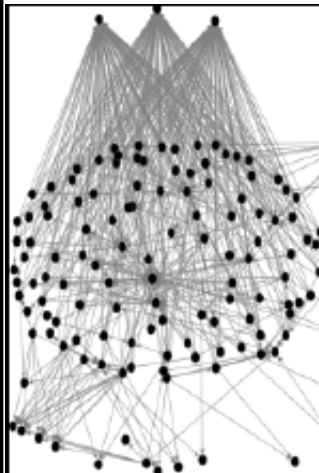
Is a [Viral upper respiratory tract infection \(disorder\)](#)
Causative agent [Virus \(organism\)](#)
Finding site [Upper respiratory tract structure \(body structure\)](#)
Pathological process [Infectious process \(qualifier value\)](#)
This concept is primitive.

Qualifiers:

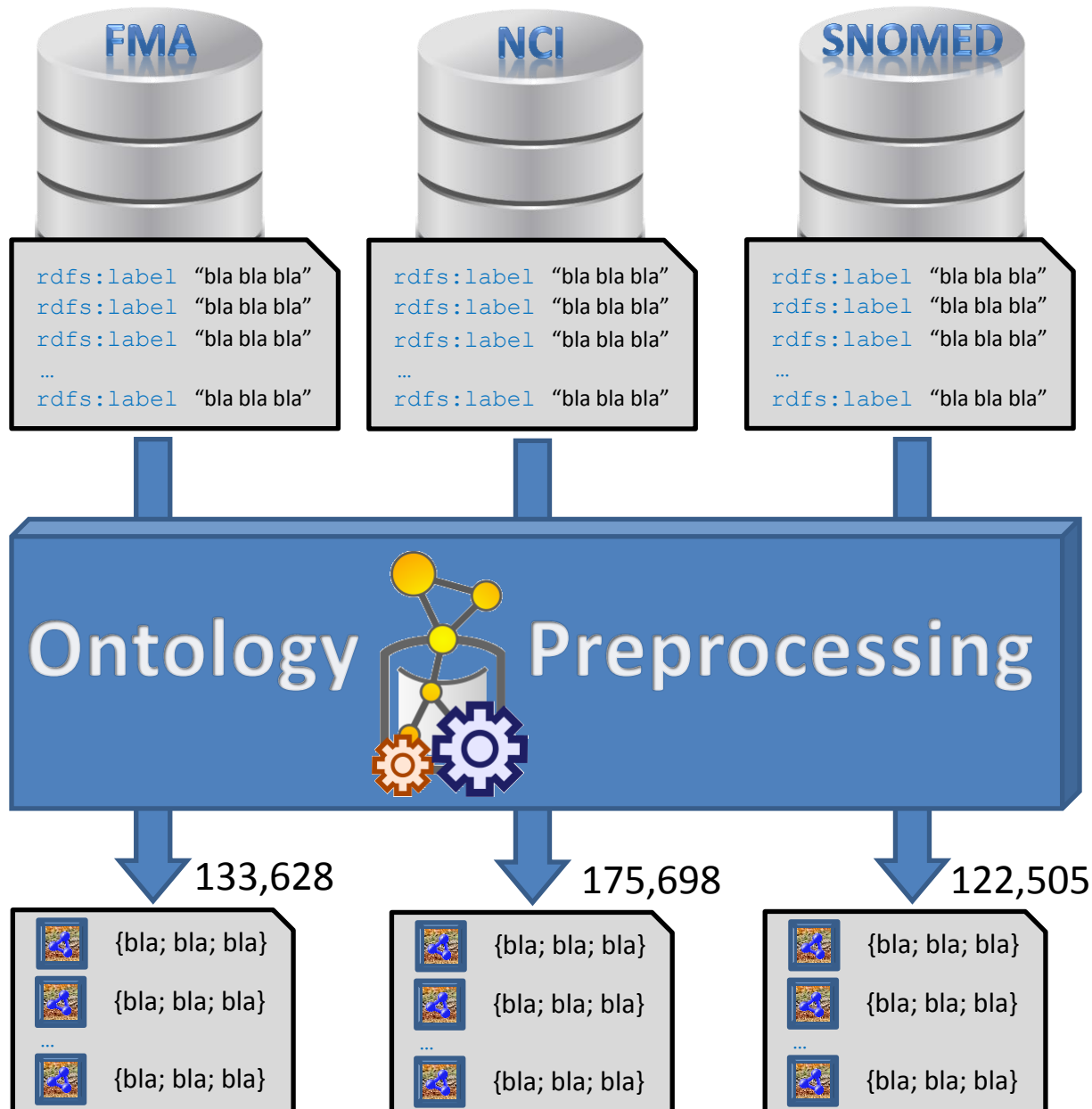
[View Qualifying Characteristics and Facts](#)

Descriptions (Synonyms):

Fully Specified Name: Common cold (disorder)
Preferred: Common cold
Synonym: Acute coryza
Synonym: Acute nasal catarrh
Synonym: Acute rhinitis
Synonym: Infective rhinitis
Synonym: Acute nasopharyngitis
Synonym: Infective nasopharyngitis
Synonym: Head cold
Synonym: Acute infective rhinitis
Synonym: Cold
Synonym: Acute nasopharyngitis, NOS
Synonym: Infective nasopharyngitis, NOS



Ontology Preprocessing (aka knowledge tokens discovery)



We preprocess the ontologies by computing as many representations for each class as it has labels (`rdfs:label`) in the ontology. For each label of each class, a set of substrings is created using all the whitespace and punctuation marks. In addition to these, we also removed possessive suffixes from the substrings and removed the 20 most common English language words according to the Oxford English Dictionary. After this preprocessing step, we end up with 133628, 175698, and 122505 representations, i.e., sets of strings for the FMA, NCI, and SNOMED ontology, respectively.

“Single Ontology – Single Tree” experiments

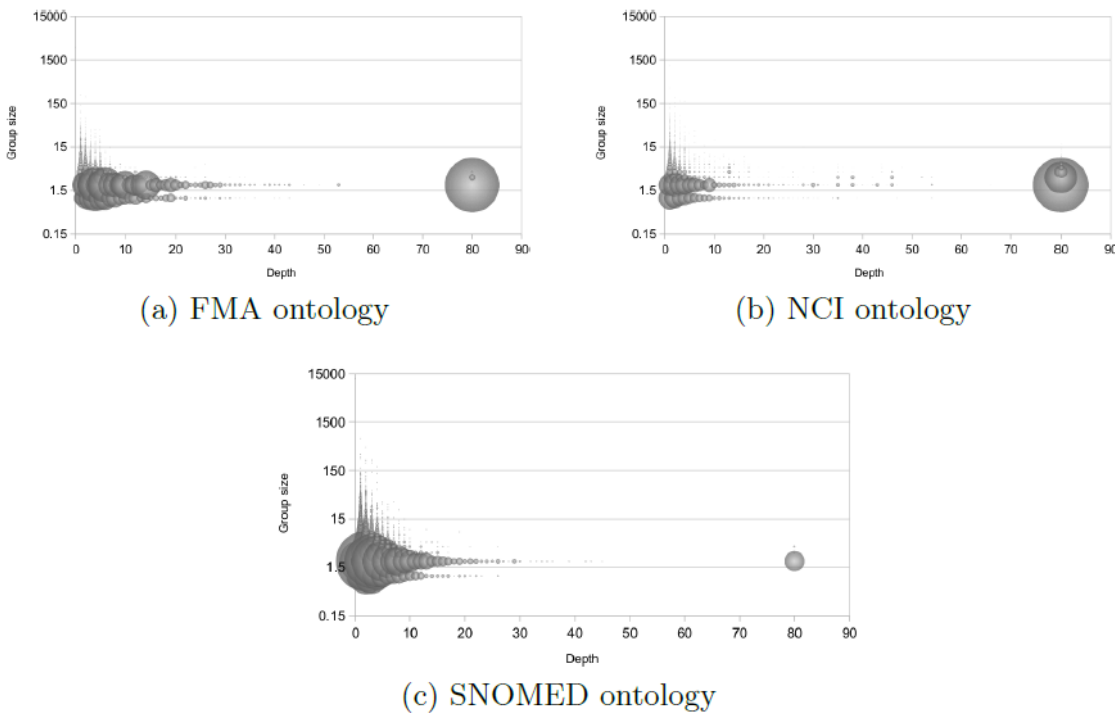


Fig : Frequency of sibling groups of a given size at a given level in one LSH Tree. Note the logarithmic scale.

In this series of experiments, we use one LSH tree and a single ontology as knowledge tokens' source. The tokenized representations are fed into the LSH tree. We then analyze the distribution of the concept representations in the tree by looking at how deep they are located in the tree and how many siblings the leaves in the tree have. Further, we investigate chains of nodes which are only there because of a low number of concept representations at the bottom of the tree.

After feeding each ontology into a single LSH Tree with $k_m = 80$, we found clusters of leaves. The figure shows how often a group of n siblings occurs as a function of the depth in the tree. What we notice is that after roughly 30 levels most of the concepts are high up in the tree. This indicates that the tree is able to distinguish between the representations fairly fast. In both the FMA and NCI ontologies, we notice a high amount of concept representations at the bottom of the tree even if k_m is chosen to be 1000, which indicates that hashing might be incapable to distinguish between the representations, i.e., they are so close that their hashes virtually always look the same.

“Processing the 2 Ontologies mix with the LSH Forest” experiments

EFFECTIVENESS

The objective was to show how processing the mix of two big datasets using LSH Forest compares to standard LSH. When mixing the SNOMED and NCI ontologies using a single tree of height 480, we obtain the precision* of 0.838, recall** of 0.547, and hence F-measure*** of 0.662. These results are similar to the results of the standard LSH algorithm which attained the precision of 0.842, recall of 0.535, and F-measure of 0.654.

COMPARABLE EFFECTIVENESS
BUT BETTER EFFICIENCY



*Precision (or correctness measure) in ontology matching measures the ratio of correctly found correspondences over the total number of returned correspondences.

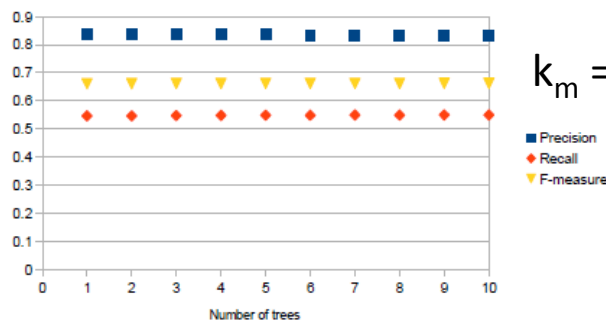
**Recall (or completeness measure) measures the ratio of correctly found correspondences over the total number of expected correspondences.

*** F-Measure is a harmonic mean of the precision and the recall.

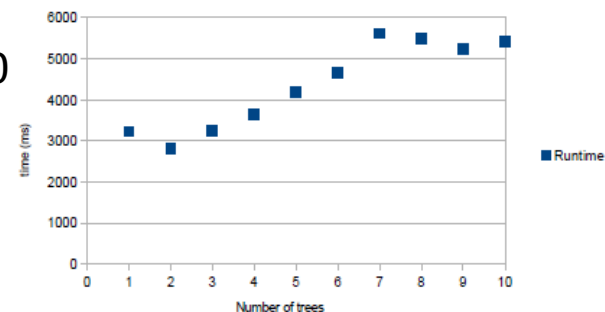
EFFICIENCY

- (a) The LSH Forest algorithm uses only 30% of the amount of hash function evaluations compared to the standard LSH algorithm.
- (b) The Forest saves 90% of the memory used for storing the result of the hash evaluations.
- (c) Using the Forest, alignment is done in 20.6 seconds, while the standard algorithm completes in 21.5 seconds.

Constraining the height of a tree does not affect the quality badly. However the tree works much faster when its height is reduced. We noticed that using the forest and better hashing, we can create a system which is roughly 7 times faster and produces results of similar quality.



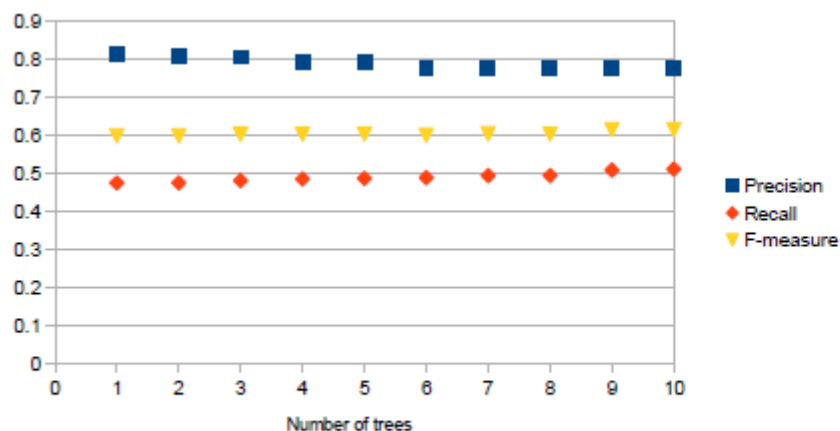
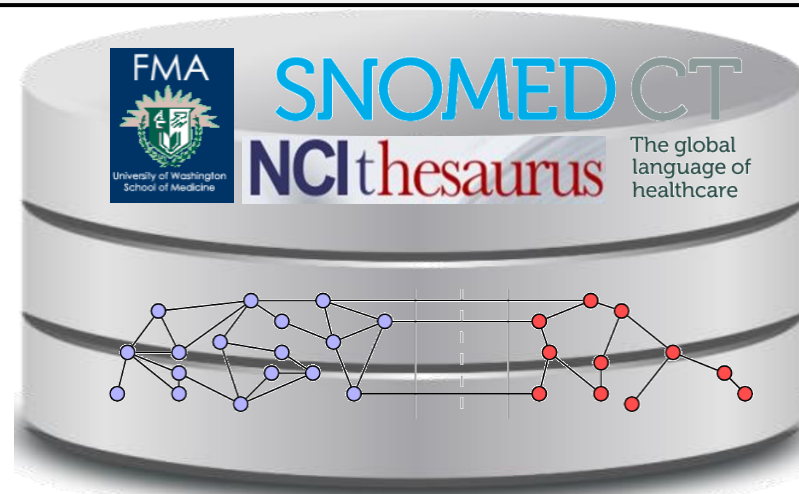
(a) Quality measurements



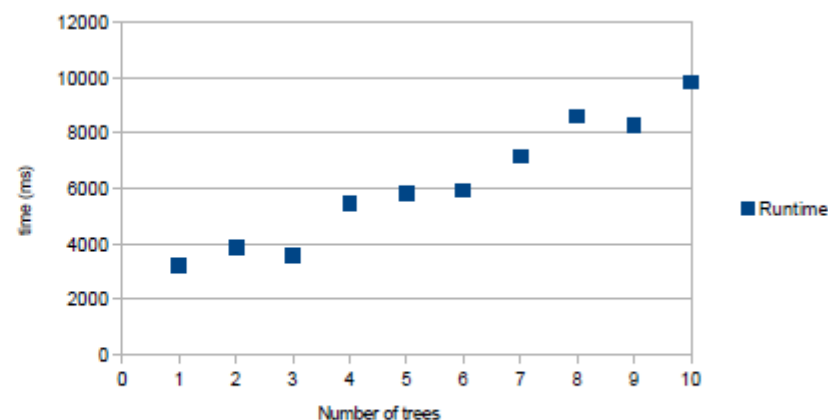
(b) Runtime

“Processing the 3 Ontologies mix with the LSH Forest” experiments

To get bigger datasets for the experiments, we feed three ontologies into the system and present similar measurements. Now, we notice the effect on the precision and recall more profoundly. Also the runtime increases faster when the input is larger. Using the LSH Forest we can perform the 3-way alignment in less than 10 seconds, with the fastest of 3.2 seconds when using a single tree, yielding roughly a ten-fold speed-up.



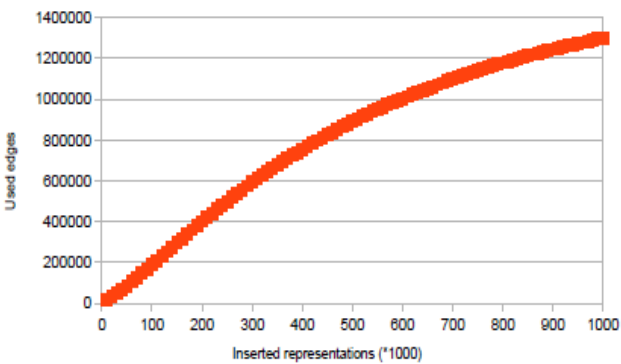
(a) Quality measurements



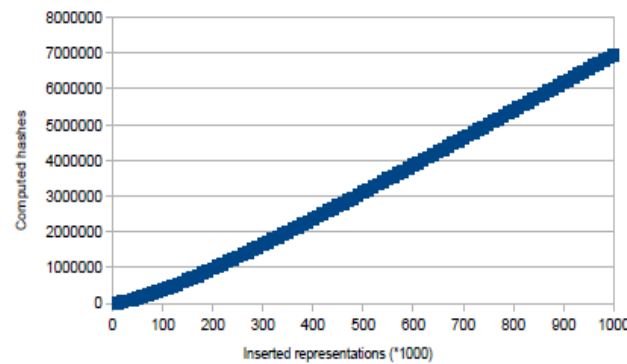
(b) Runtime

“Adding Dynamics” to the experiments (uniform distribution case)

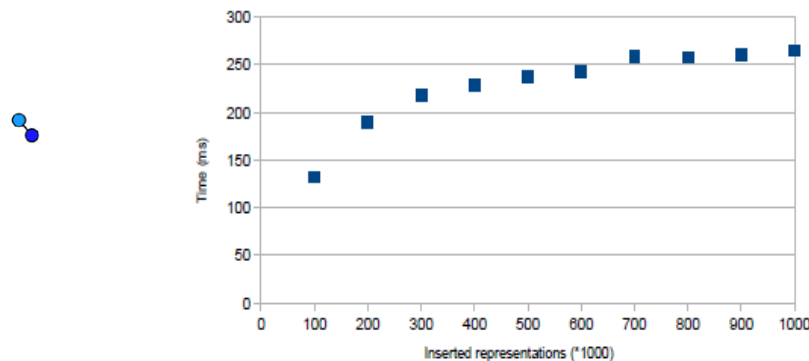
In the final series of experiments we observe how the tree reacts to dynamic (incremental) insertion of concepts. In the basic case, we select 10^6 classes from the three ontologies using a uniform distribution. The class representations are then one by one inserted into the tree and after every 10^4 insertions we observe the cumulative memory consumption and processing need by measuring and number of edges and number of hashing operations used. We also measure the time needed for insertion of every 10^5 concept representations.



(a) Edges



(b) Hashes

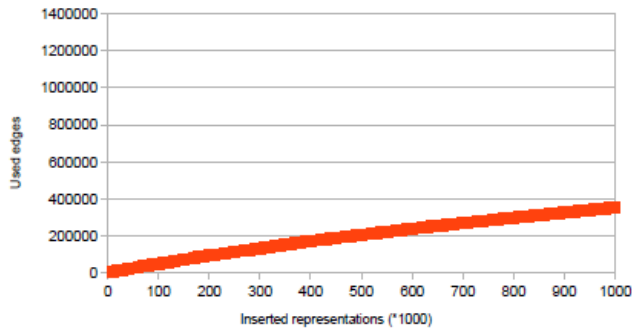


(c) Time needed to insert the previous 100000 concepts

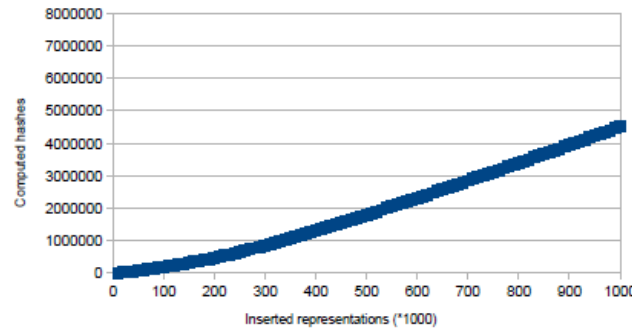
We note that the number of edges needed grows sub-linear. This is as expected since both the fact that certain concept representation will be selected more as once and the reuse of edges decreases the number of new edges needed. The number of hashes shows an initial ramp-up and then starts growing linear. We also note that the time used for adding is growing, but the growth slows down when more concepts are added. Moreover, if we try to fit a linear curve through the cumulative runtime measurements, we notice that we can obtain a Pearson product-moment correlation coefficient of 0.9976, indicating that the increase is actually very close to linear.

“Adding Dynamics” to the experiments (Zipf distribution case)

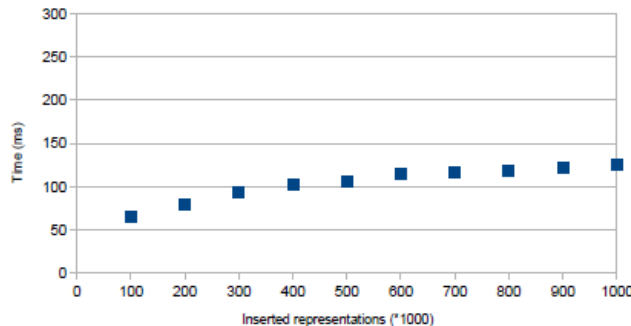
When choosing the representations using a Zipf distribution we obtain the results as depicted in the figure.



(a) Edges



(b) Hashes

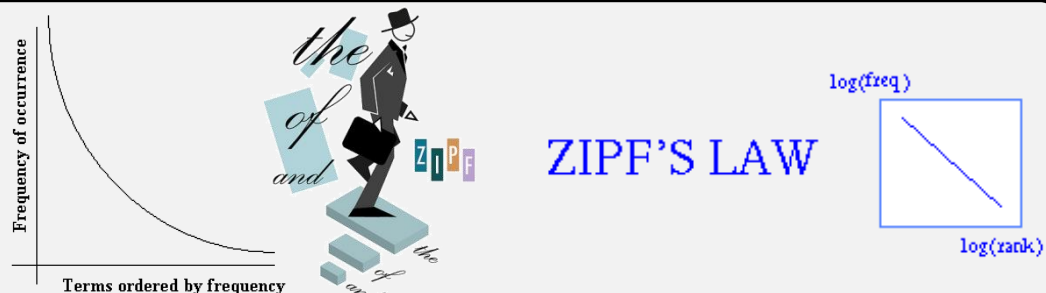


(c) Time needed to insert the previous 100000 concepts

Zipf's law states that given some corpus of natural language utterances, the frequency of any word is inversely proportional to its rank in the frequency table. Thus the most frequent word will occur approximately twice as often as the second most frequent word, three times as often as the third most frequent word, etc.

In our case, according to Zipf distribution, there will be concepts which are inserted much more often than others. Some will never be inserted at all. With the uniform distribution all the concepts are inserted with equal probability.

When comparing the charts for insertion using the normal and Zipf distribution, we notice that the later puts much less of a burden upon the system. This is a desirable effect since it means that the system is likely to work well with more organic loads. Also here, we can draw a linear curve trough the cumulative runtime measurements with a high correlation coefficient of 0,9968.



Conclusions

- When trying to understand and follow what is happening around us, we have to be able to connect different pieces of information together. The amount of information which we perceive does not allow us to look at each detail (suffered effectiveness); instead we need to focus on specific parts and ignore the rest (aiming the efficiency). When we want to build a system capable of embodying evolution in knowledge, similar challenges have to be tackled. We investigated how to bring related pieces of knowledge together while balancing between the effectiveness and the efficiency.
- We looked at the application of LSH Forest to dynamically sow knowledge tokens in the environmental contexts. The problem is formally regarded as a dynamic ontology matching problem.
- We found out that LSH Forest is a suitable approach because it is able to balance well between efficiency and effectiveness. This can be observed from the fact that the method scales well, both from a space and runtime perspective; and from the fact that the quality measures are sufficiently high.

