OpenCog, AtomSpace and Structured Language Learning

Anton Kolonin
anton@singularitynet.io

Novosibirsk State University
*THE REAL SCIENCE

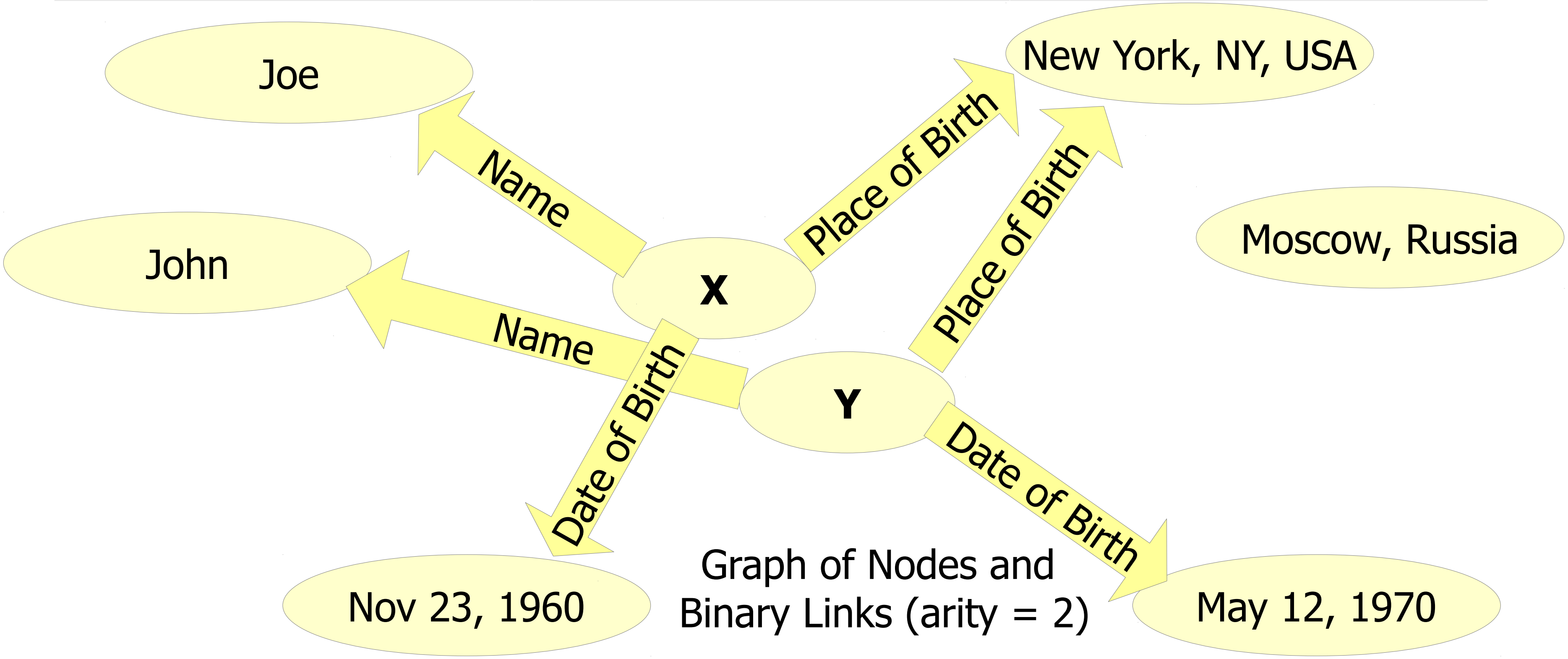
AIGENTS®
https://aigents.com

SingularityNET
https://singularitynet.io

Relational Database vs. Graph

Table of relations in relational database

Name	Date of Birth	Place of Birth
Joe	May 12, 1970	New York, NY, USA
John	Nov 23, 1960	New York, NY, USA



Graph of Nodes and Binary Links (arity = 2)

Graph as Relational Database

Nodes

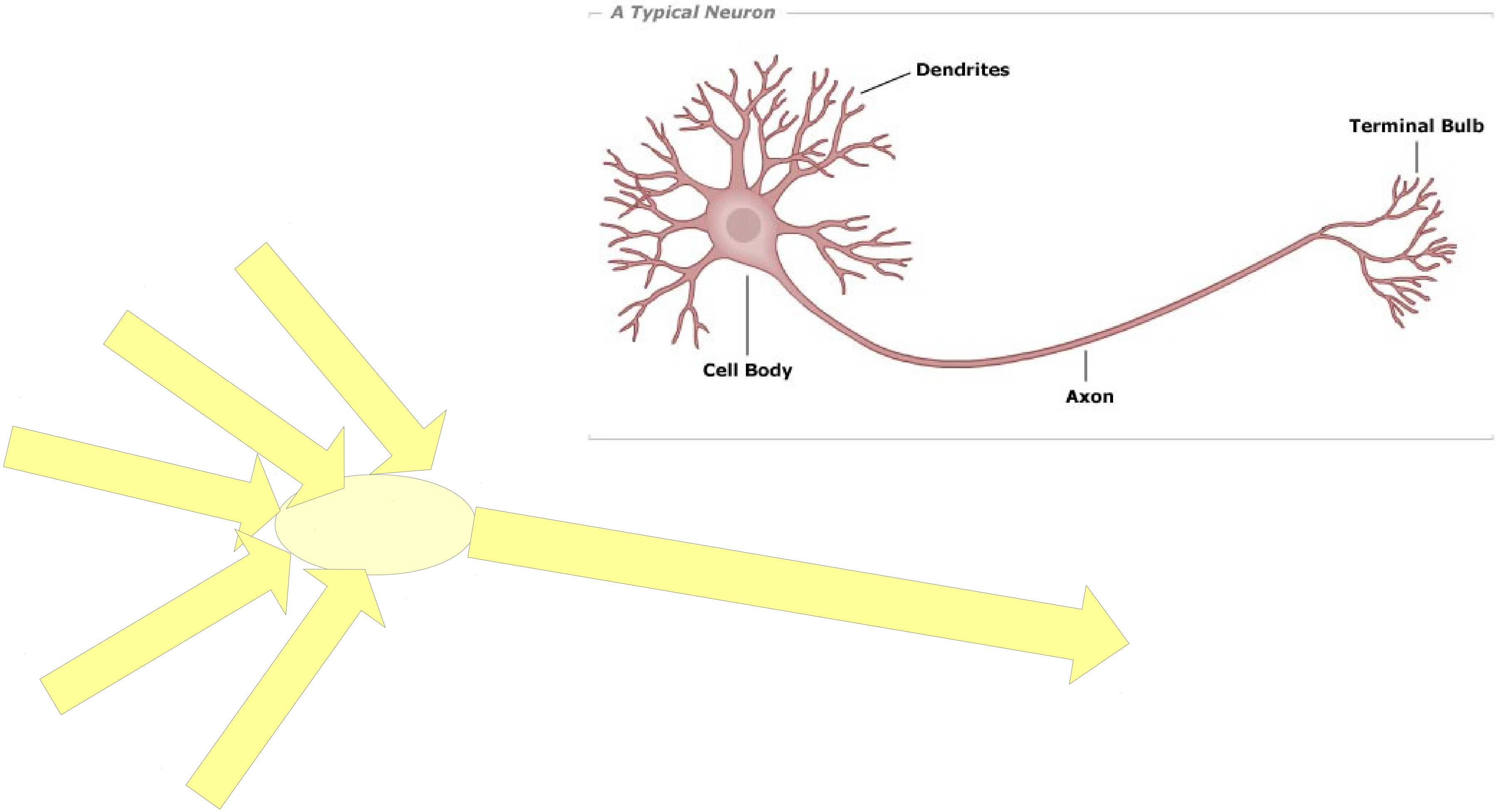
Id	Label
1	X
2	Y
3	John
4	Joe
5	Nov 23, 1960
6	May 12, 1970
7	New York, NY, USA
8	Moscow, Russia

Binary Links

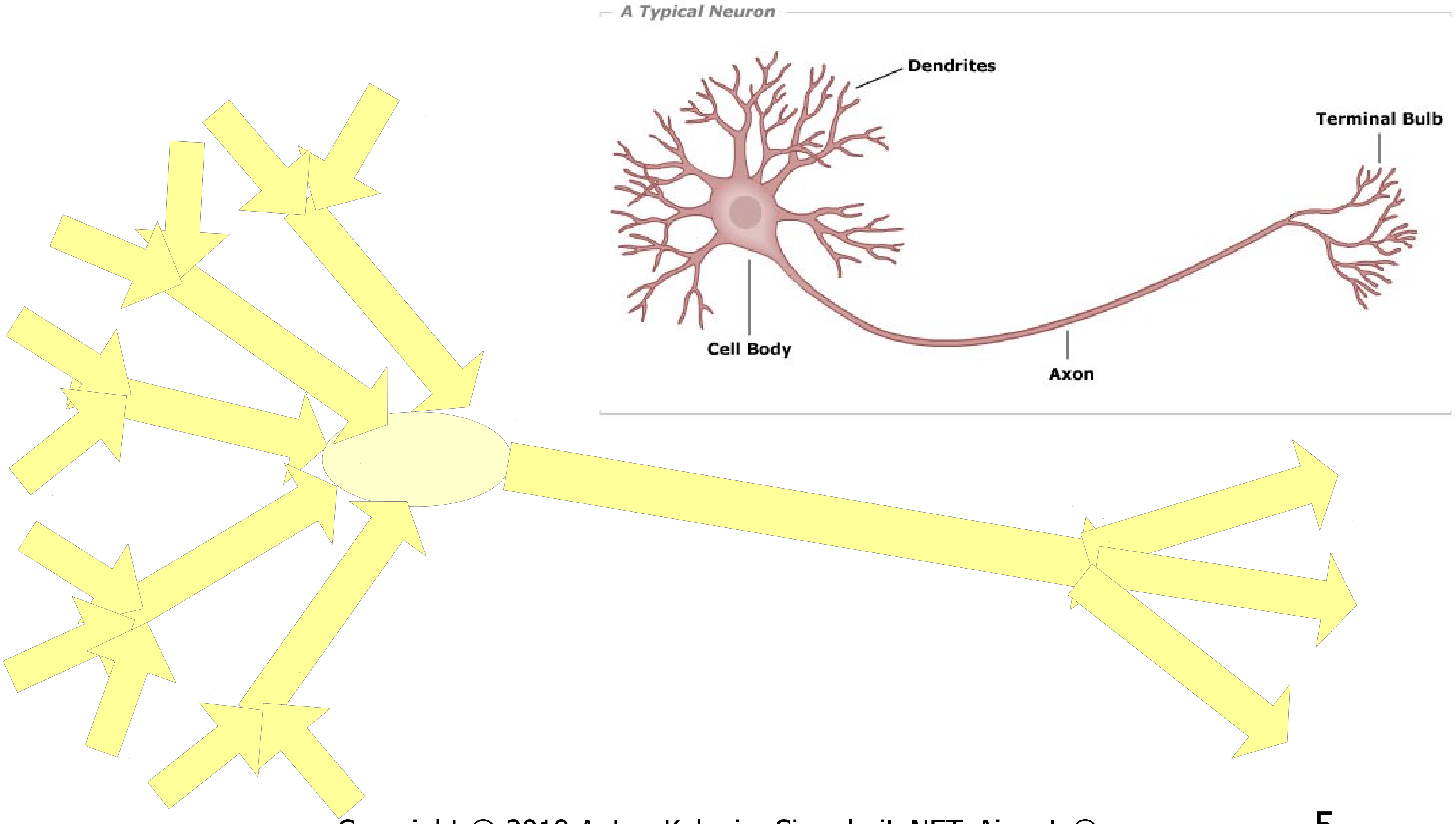
From	To	Label
1	4	Name
2	3	Name
1	5	Date of Birth
2	6	Date of Birth
1	7	Place of Birth
1	7	Place of Birth

Binary links in relational database are connecting relations while tables holding these links are called relationships

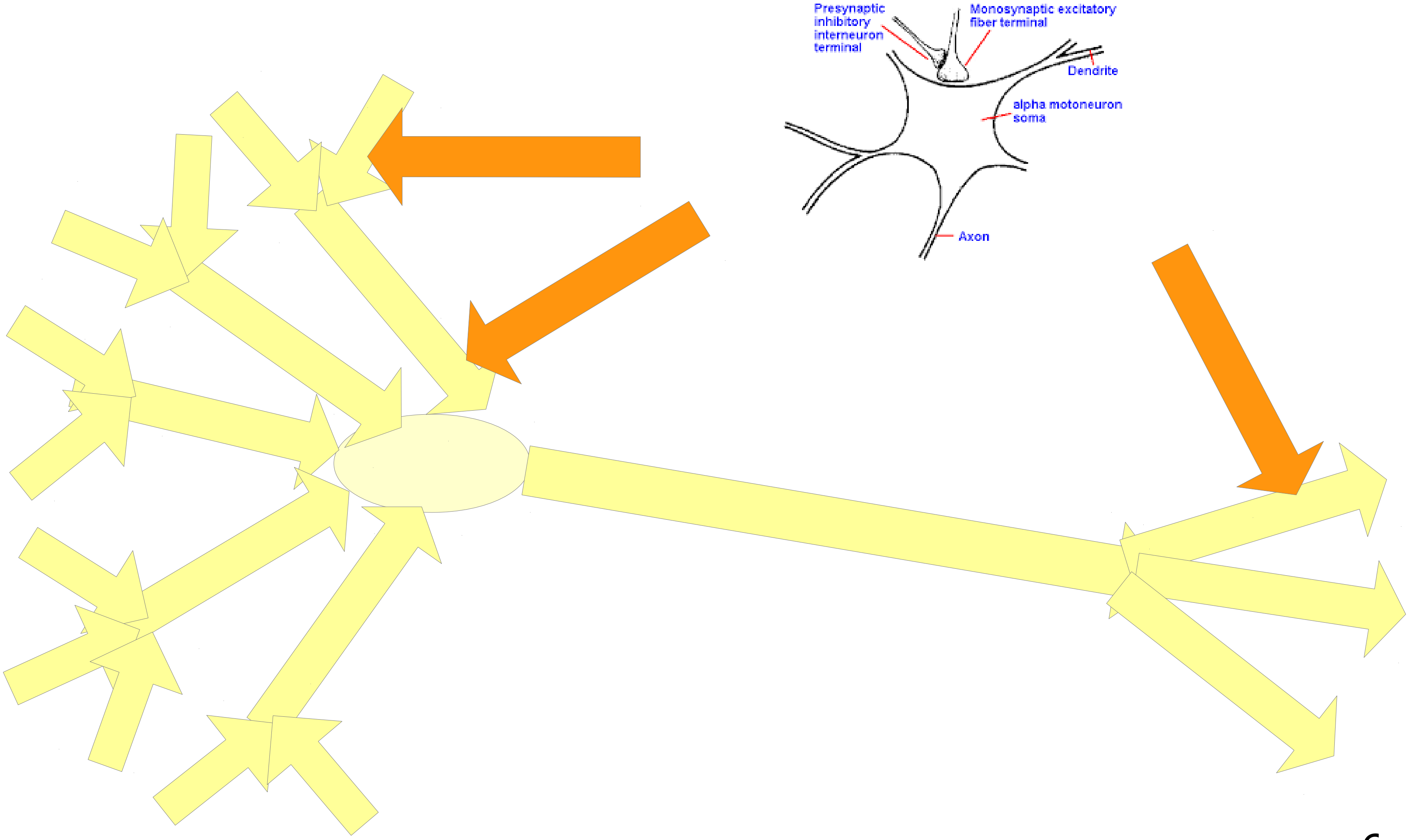
Neuron vs. Graph: Node and Links



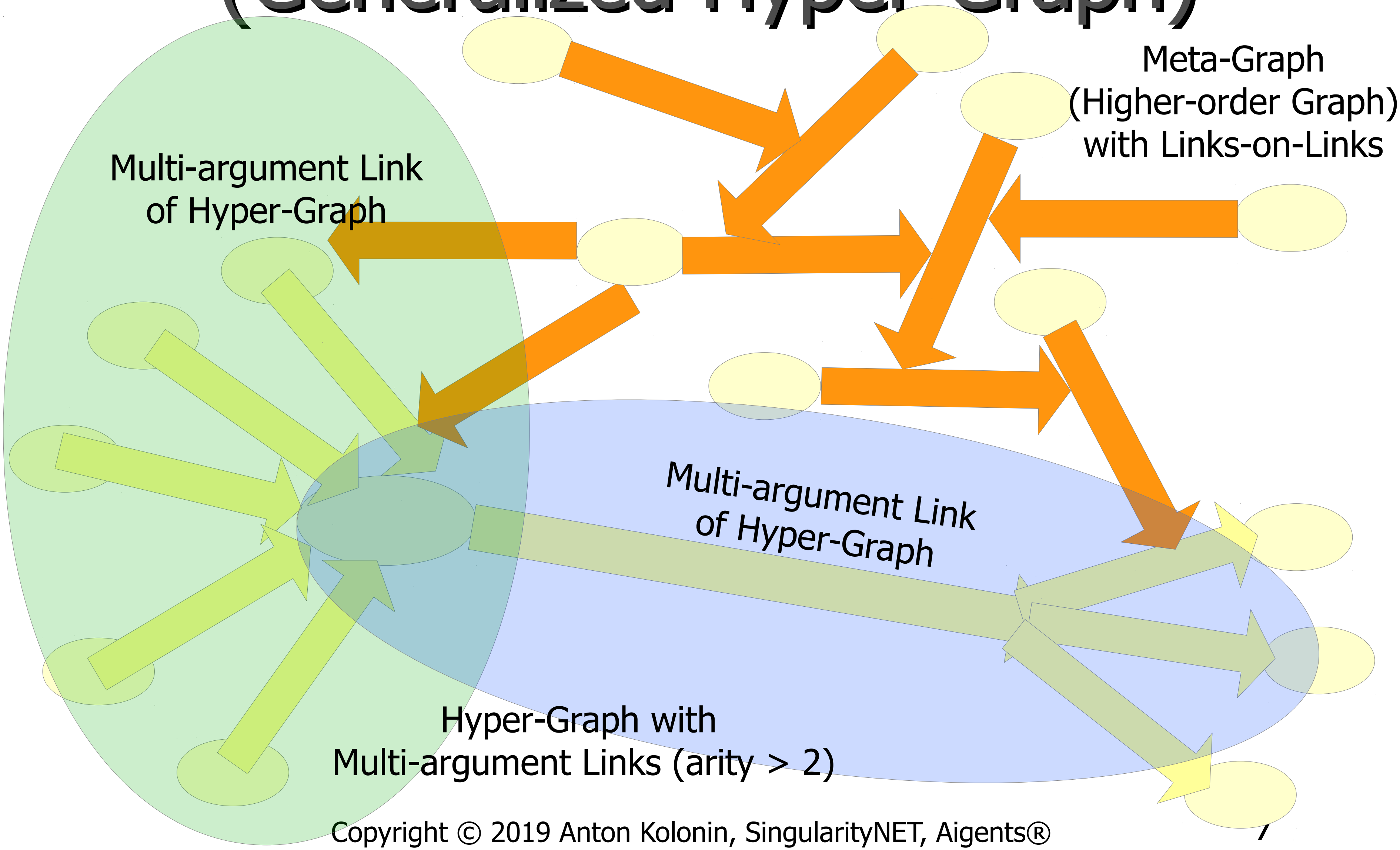
Neuron vs.Graph:Dendrites & Axons - Multi-argument Links (arity > 2)



Neuron vs. Graph: Synapses on Synapses – Links-on-Links


Presynaptic inhibitory interneuron terminal
Monosynaptic excitatory fiber terminal
Dendrite
alpha motoneuron soma
Axon

Hyper-Graph and Meta-Graph (Generalized Hyper-Graph)



Meta-Graph
(Higher-order Graph)
with Links-on-Links

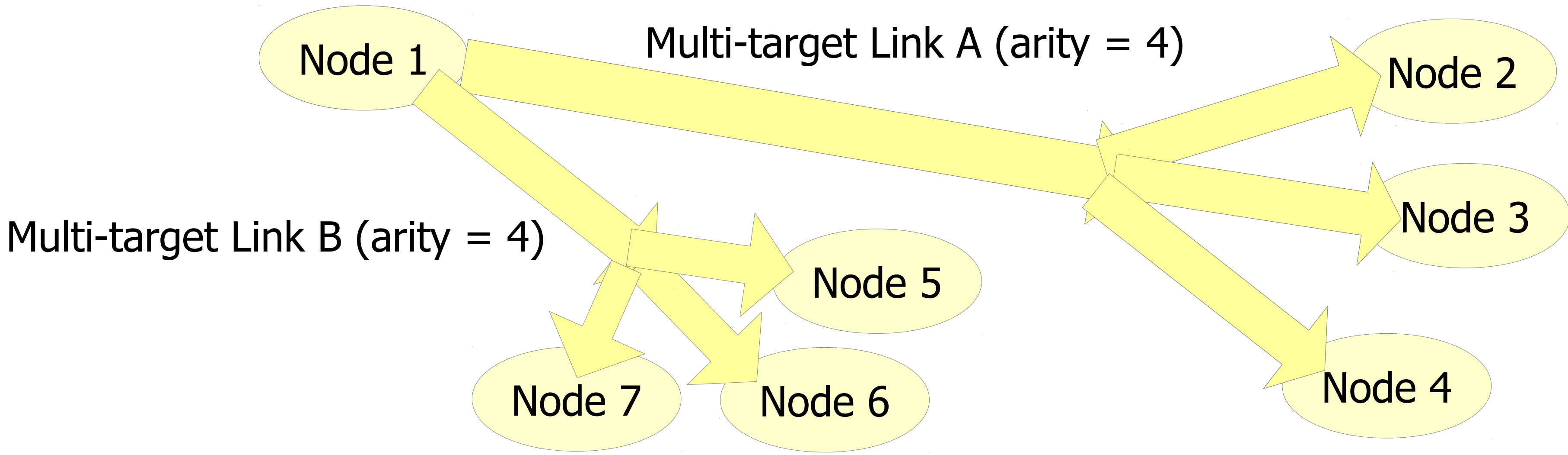
Multi-argument Link
of Hyper-Graph

Multi-argument Link
of Hyper-Graph

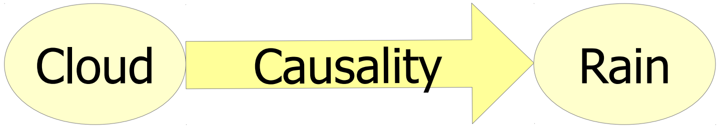
Hyper-Graph with
Multi-argument Links (arity > 2)

N-ary Multi-argument Links as N-ary Relations in Relational database

	Argument 1	Argument 2	Argument 3	Argument 4
Multi-target Link A (arity = 4)	Node 1 source	Node 2 target	Node 3 target	Node 4 target
Multi-target Link B (arity = 4)	Node 1 source	Node 5 target	Node 6 target	Node 7 target


Node 1
Multi-target Link A (arity = 4)
Node 2
Node 3
Multi-target Link B (arity = 4)
Node 5
Node 7
Node 6
Node 4

Ordered (directed) and Unordered (undirected) Links

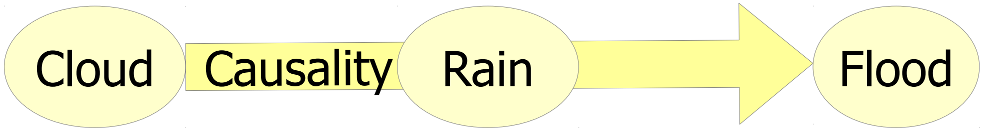


Cloud

Causality

Rain

Ordered (directed) Link, arity = 2
Cloud is “source”
Rain is “target”
Real Link



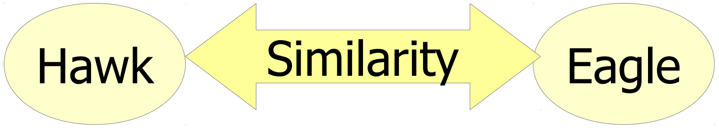
Cloud

Causality

Rain

Flood

Ordered (directed) Link, arity = 3
Cloud is “source”
Rain is “source” and “target”
Flood is “target”
Sequence or Series

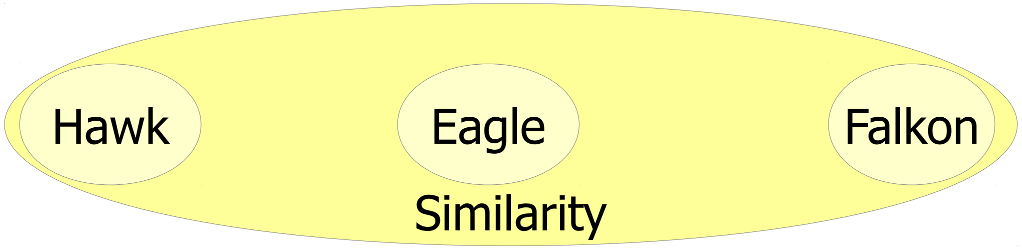


Hawk

Similarity

Eagle

Unordered (undirected) Link, arity = 2
Just Pair



Hawk

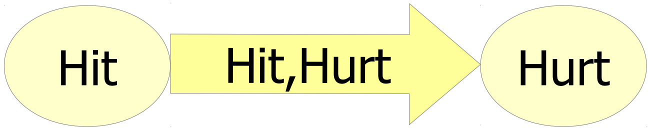
Eagle

Falkon

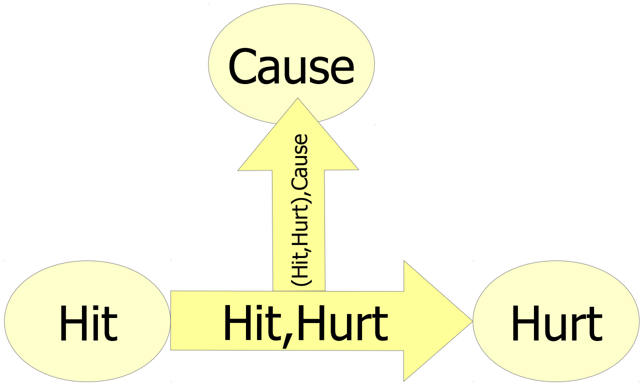
Similarity

Unordered (undirected) Link, arity = 3
Set

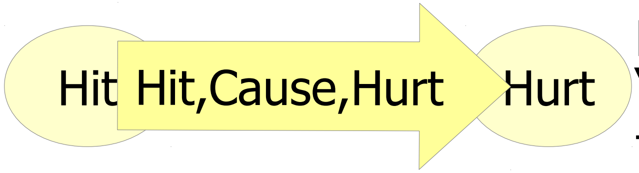
Link Type and Arity – ways to go



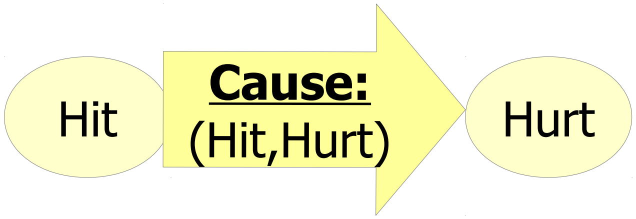
Binary typeless link (from, to) with arity = 2



Binary link (from, to) with arity = 2 where type is expressed with extra binary link to link type node (needs higher-order links-on-links capability)



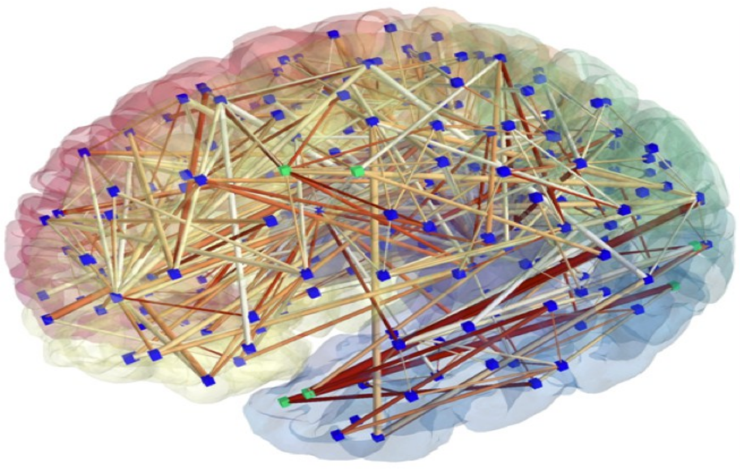
Ternary triple (from, type, to) with arity = 3 where type is part of relation, corresponds to “term” subject-predicate-object in natural language processing and “triple” in most of conventional graph databases (RDF triplet stores)

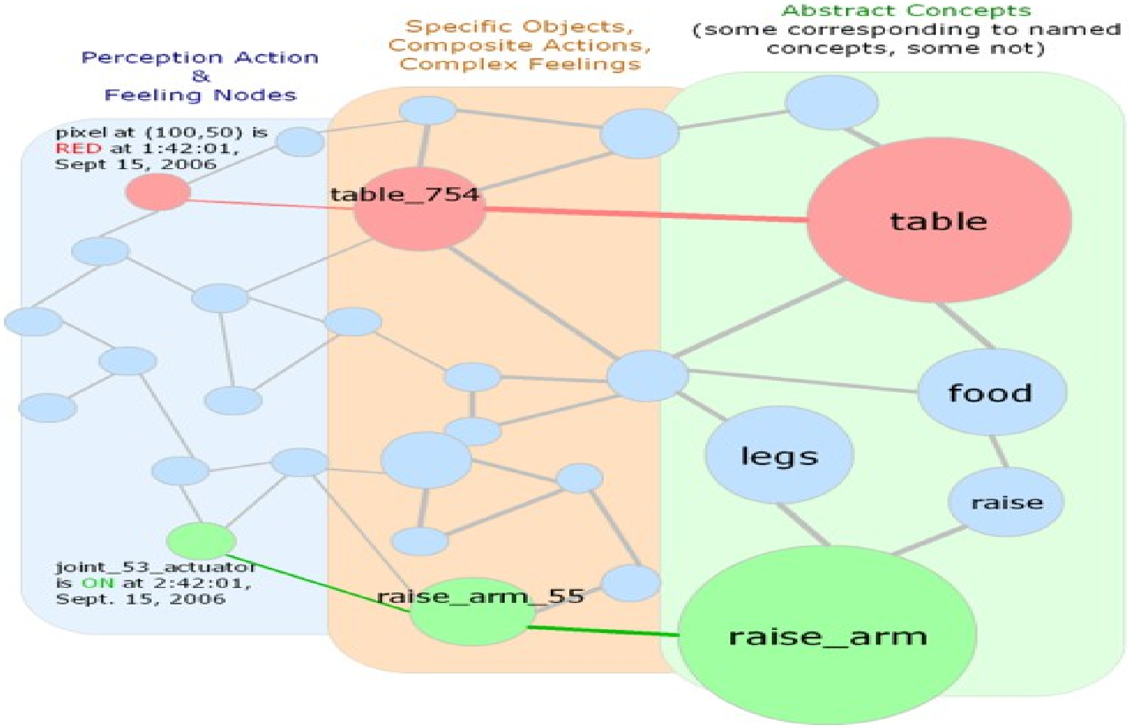


Separate predicate (type) and list of its arguments (from, to) with arity = 2 corresponds to “term” in predicate logic and RDF statement in modern semantic web programming and implemented in OpenCog and Webstructor, dealing with arbitrary arities, specific to those predicate types

OpenCog’s “AtomSpace”

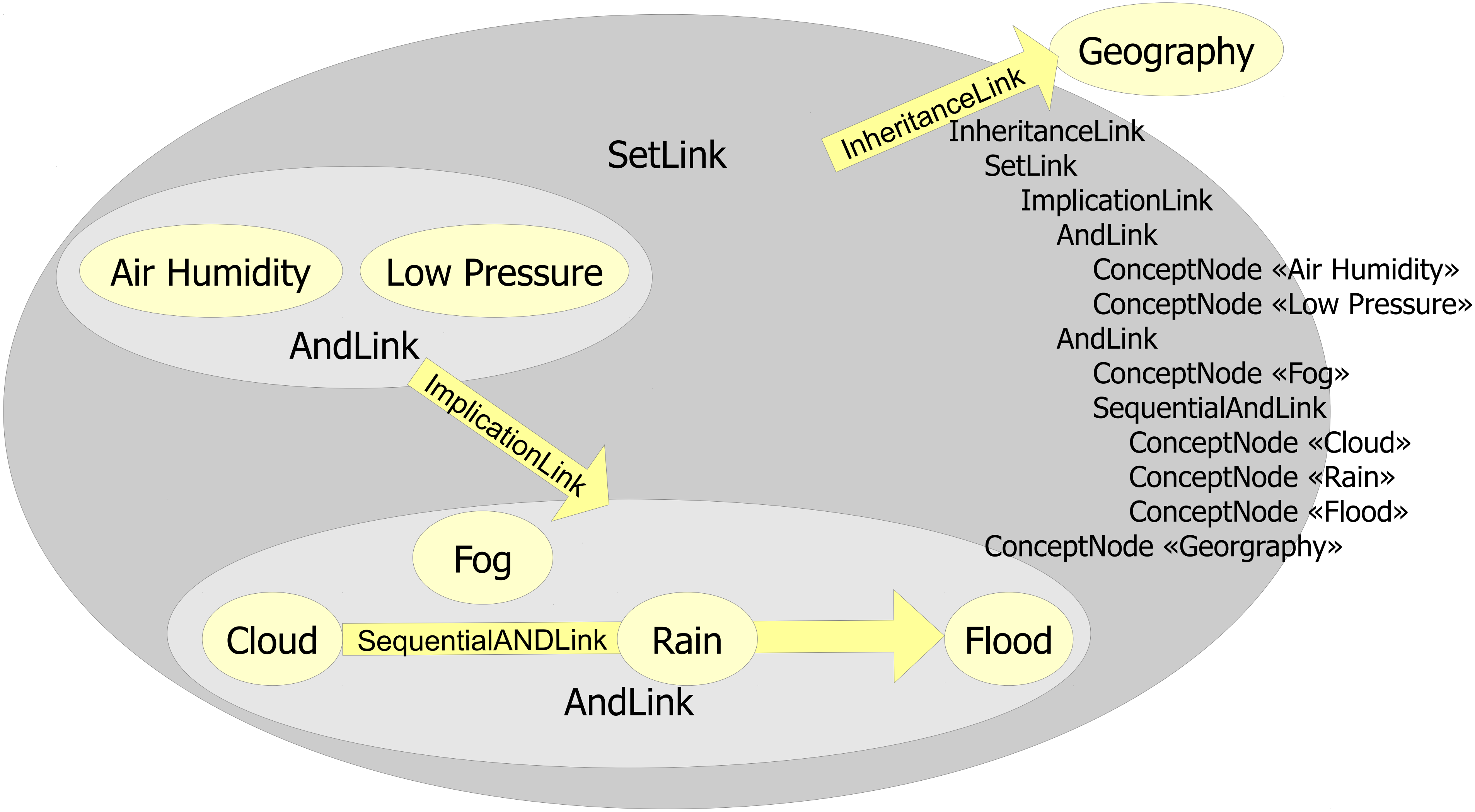
Implements Generalized Hyper-graph and Meta-graph, so each directed/undirected link may link together any number of atoms, where atom could be either node (arity = 0) or any other link with any arity, including unordered N-ary links representing subgraphs as their elements.




Perception Action & Feeling Nodes
Specific Objects, Composite Actions, Complex Feelings
Abstract Concepts (some corresponding to named concepts, some not)
pixel at (100,50) is RED at 1:42:01, Sept 15, 2006
table_754
table
food
legs
raise
joint_53_actuator is ON at 2:42:01, Sept. 15, 2006
raise_arm_55
raise_arm

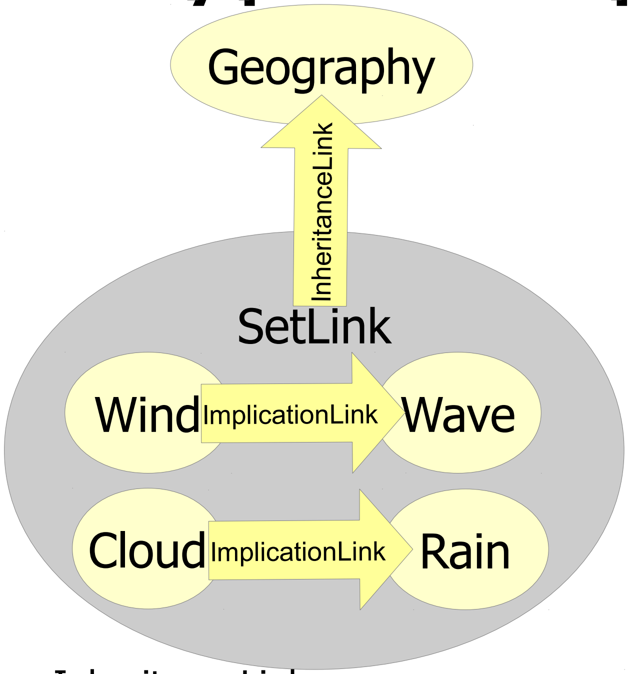
https://github.com/opencog/atomspace

AtomSpace - Generalized (Link-as-Node) Hyper-Graph is a Meta-Graph (in Atomese)

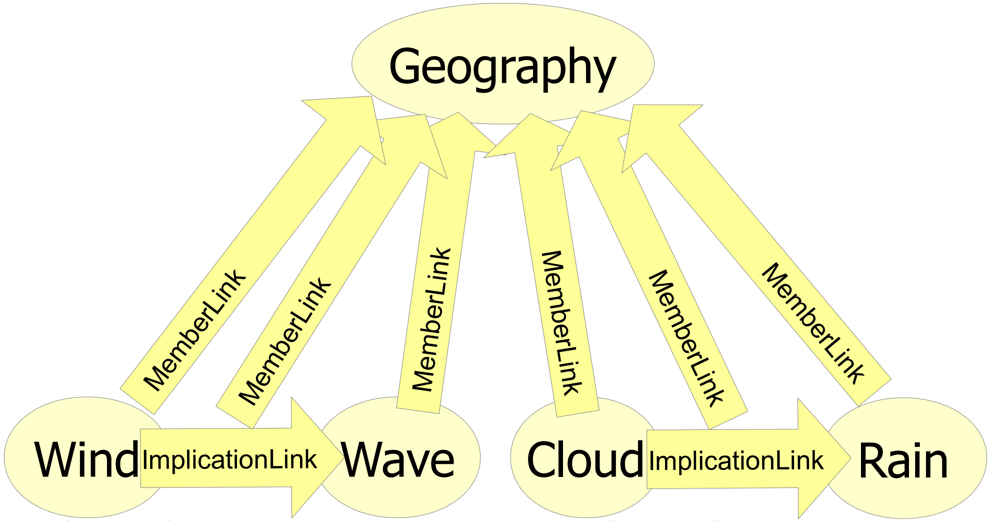


```
InheritanceLink
  SetLink
    ImplicationLink
      AndLink
        ConceptNode «Air Humidity»
        ConceptNode «Low Pressure»
      AndLink
        ConceptNode «Fog»
        SequentialAndLink
          ConceptNode «Cloud»
          ConceptNode «Rain»
          ConceptNode «Flood»
  ConceptNode «Georgraphy»
```

AtomSpace - Meta-Graphs with Hyper-Graphs and Links-on-Links



```
InheritanceLink
    SetLink
        ImplicationLink
            ConceptNode «Wind»
            ConceptNode «Wave»
        ImplicationLink
            ConceptNode «Cloud»
            ConceptNode «Rain»
      ConceptNode «Georgraphy»
```



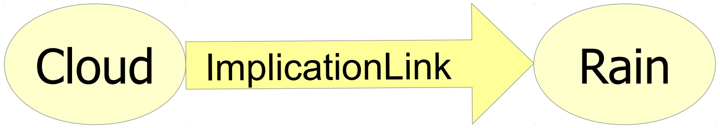
```
MemberLink
     ConceptNode «Wind»
     ConceptNode «Georgraphy»
MemberLink
     ConceptNode «Wave»
     ConceptNode «Georgraphy»

MemberLink
     ImplicationLink
         ConceptNode «Wind»
         ConceptNode «Wave»
     ConceptNode «Georgraphy»
```

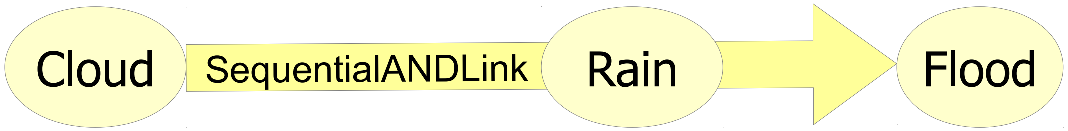
```
MemberLink
     ConceptNode «Cloud»
     ConceptNode «Georgraphy»
MemberLink
     ConceptNode «Rain»
     ConceptNode «Georgraphy»

MemberLink
     ImplicationLink
         ConceptNode «Cloud»
         ConceptNode «Rain»
     ConceptNode «Georgraphy»
```

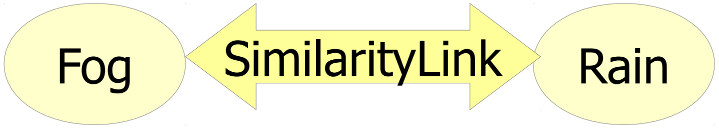
Ordered (directed) and Unordered (undirected) Links



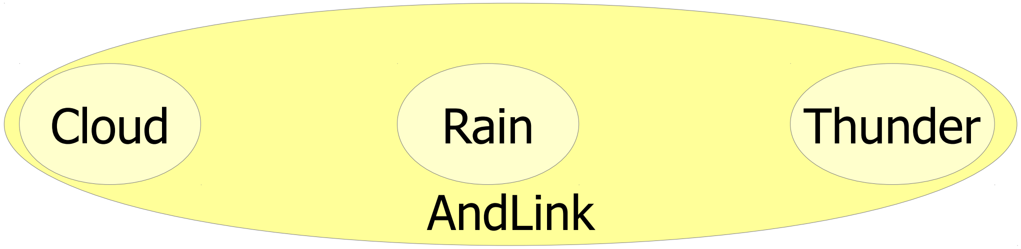
```
ImplicationLink
    ConceptNode ”Cloud”
    ConceptNode ”Rain”
```



```
SequentialANDLink
    ConceptNode ”Cloud”
    ConceptNode ”Rain”
    ConceptNode ”Flood”
```



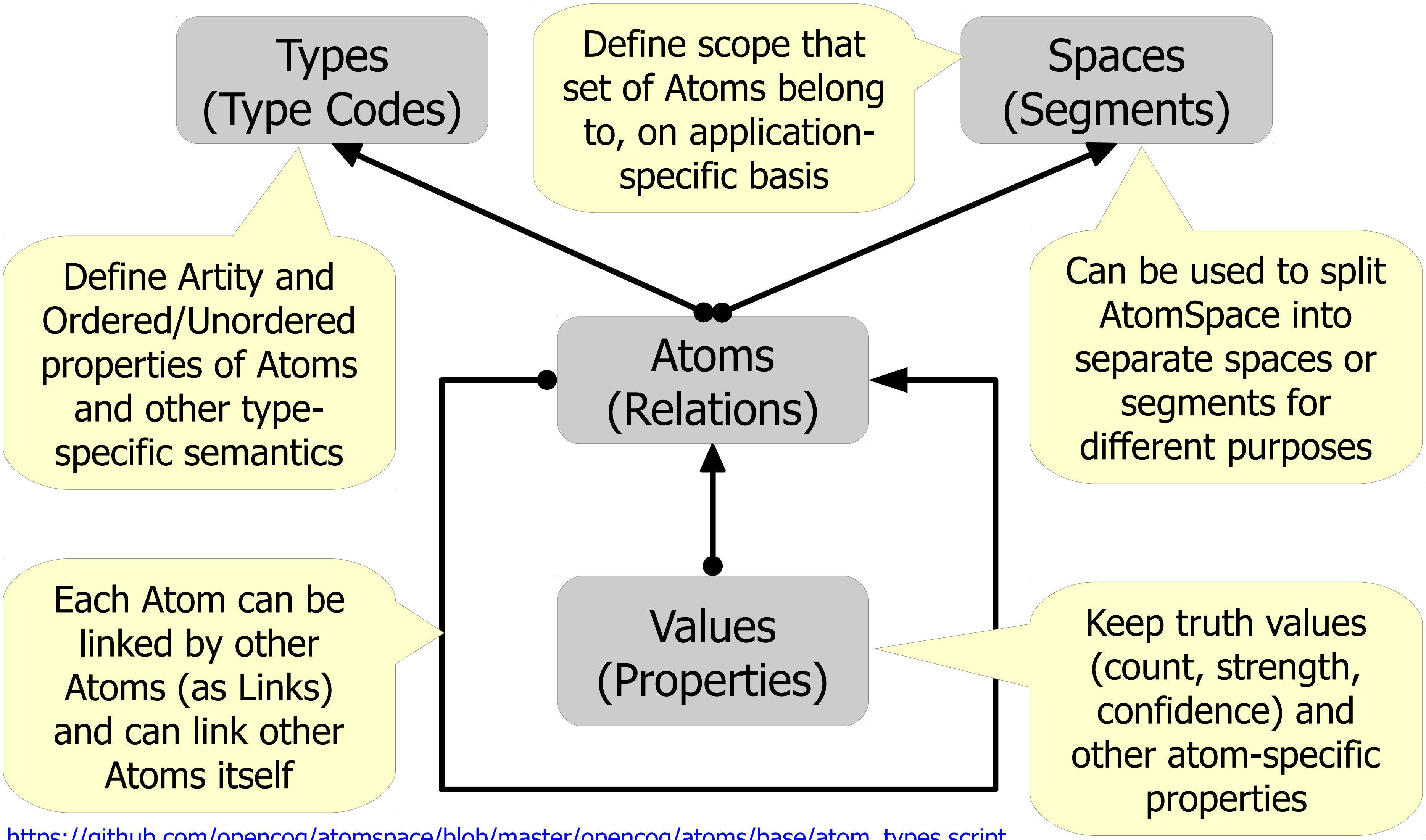
```
SimilarityLink
    ConceptNode ”Fog”
    ConceptNode ”Rain”
```



```
AndLink
    ConceptNode ”Cloud”
    ConceptNode ”Rain”
    ConceptNode ”Thunder”
```

expressed in Atomese language

OpenCog AtomSpace, what’s inside



Types (Type Codes)

Define Artity and Ordered/Unordered properties of Atoms and other type-specific semantics

Define scope that set of Atoms belong to, on application-specific basis

Spaces (Segments)

Can be used to split AtomSpace into separate spaces or segments for different purposes

Atoms (Relations)

Each Atom can be linked by other Atoms (as Links) and can link other Atoms itself

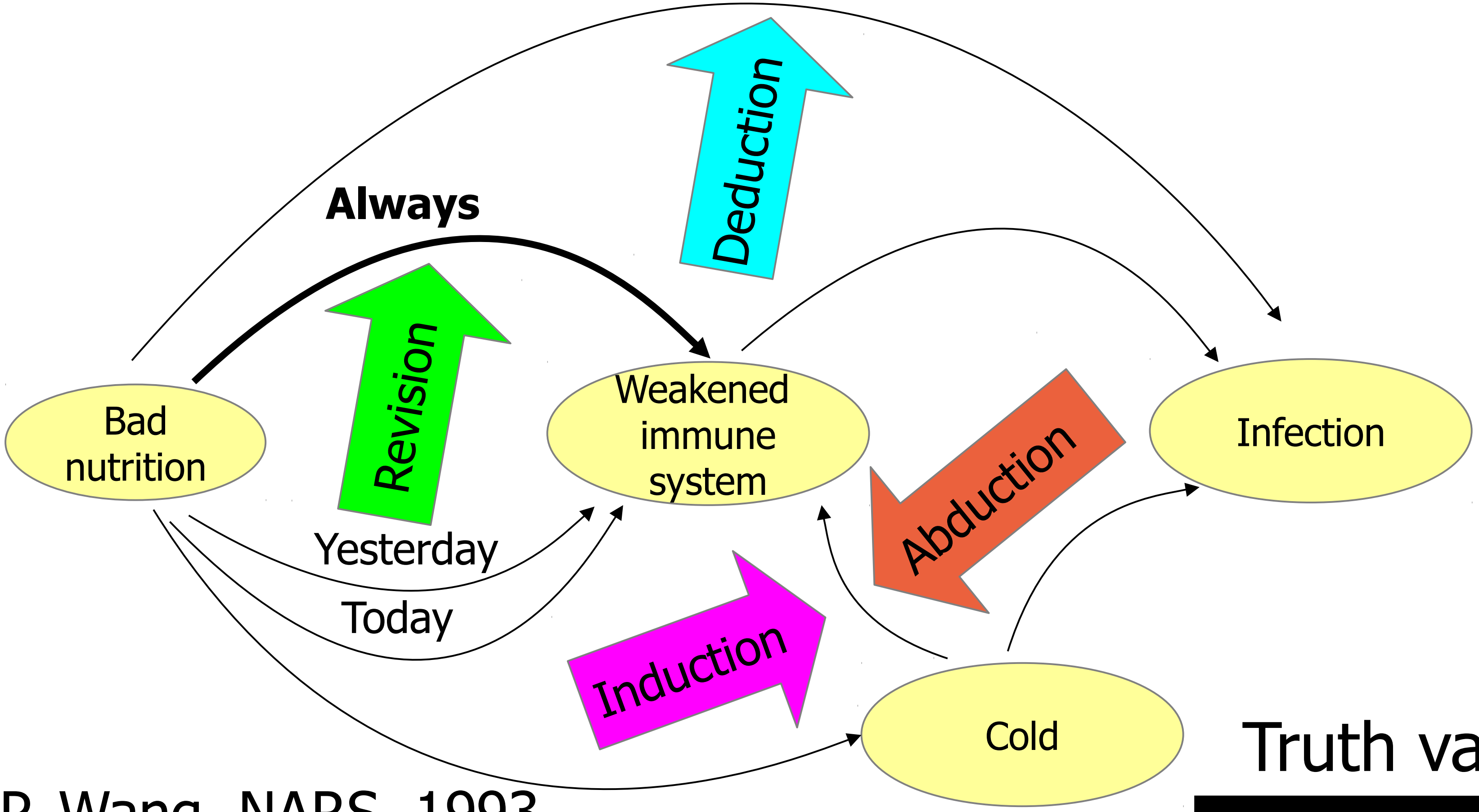
Values (Properties)

Keep truth values (count, strength, confidence) and other atom-specific properties

https://github.com/opencog/atomspace/blob/master/opencog/atoms/base/atom_types.script

Complex Truth Values for Probabilistic Logic

Example: Non-Axiomatic Reasoning System (NARS)



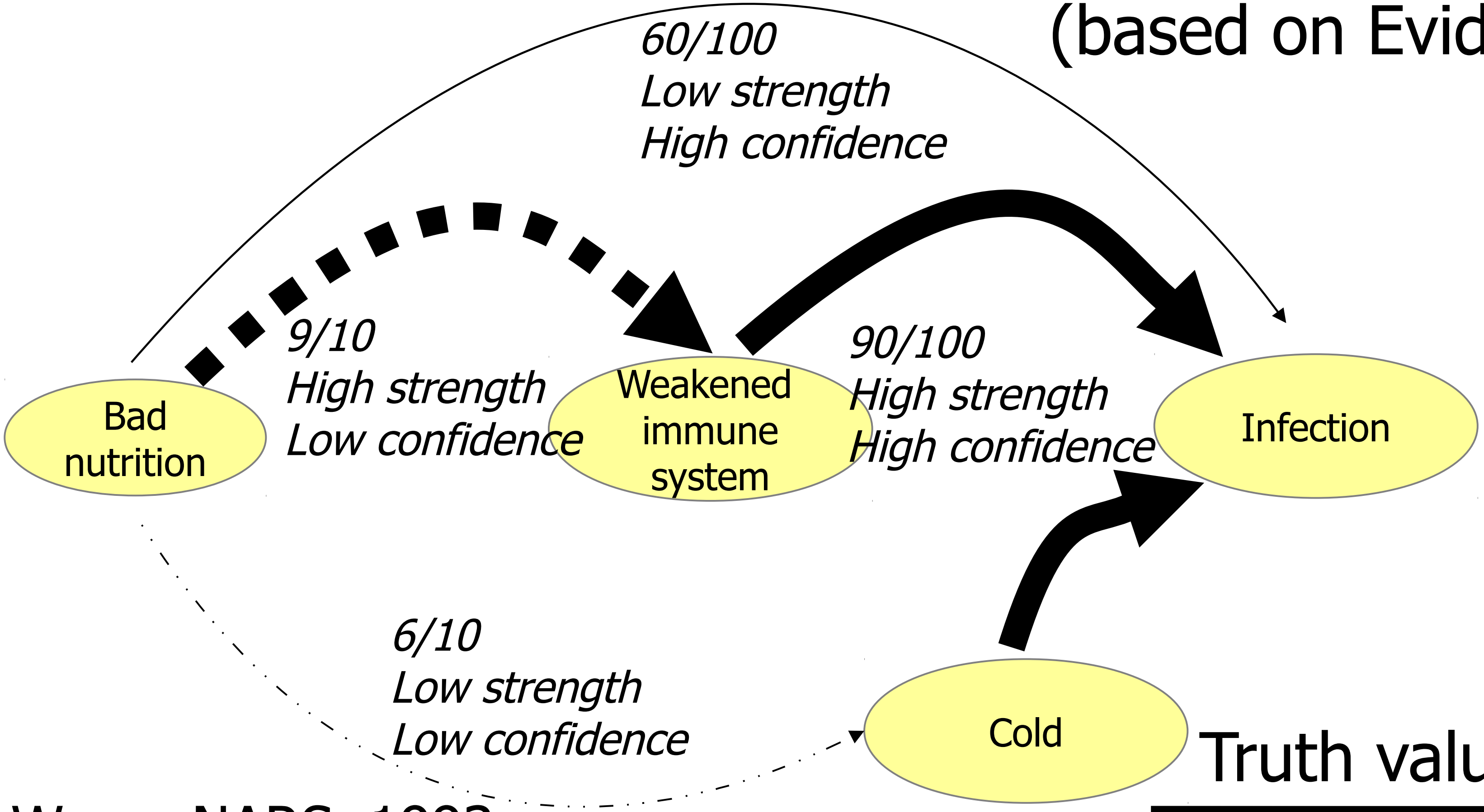
Truth value:

Strength, Confidence

P. Wang, NARS, 1993
E. Vityaev, Discovery, 2001
B. Goertzel et al., PLN, 2008

Complex Truth Values for Probabilistic Logic

Probabilistic Logic: Separating Strength and Confidence (based on Evidence)



P. Wang, NARS, 1993
E. Vityaev, Discovery, 2001
B. Goertzel et al., PLN, 2008

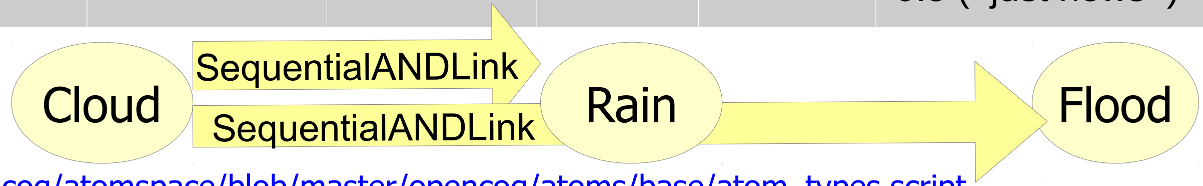
Truth value:

Strength, Confidence

OpenCog Atoms - example

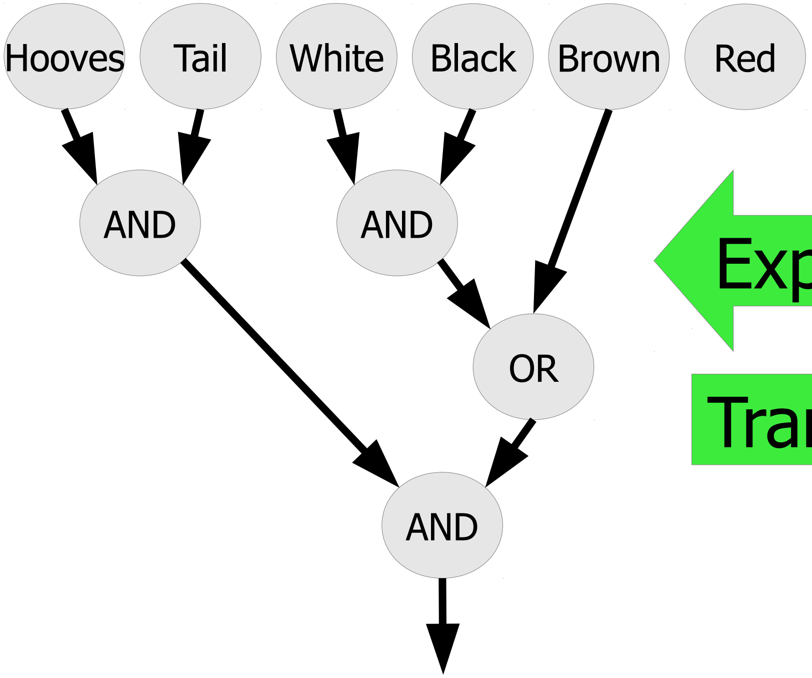
Id	Type	Space	Name (Label)	Level in Meta-Graph	Arity	Argu-ments	Truth Value: Count (Evidence), Strength, Confidence
11	SequentialANDLink	5	-	2	2	13, 14	22 0.5 (22/44=50%) 1.0 («surely»)
12	SequentialANDLink	5	-	2	3	13, 14, 15	11 0.25 (11/44=25%) 0.5 («probably»)
13	ConceptNode	5	«Cloud»	0	0	-	44 1.0 1.0 («fact»)
14	ConceptNode	5	«Rain»	0	0	-	22 1.0 1.0 («fact»)
15	ConceptNode	5	«Flood»	0	0	-	11 1.0 0.5 («just news»)

Atom Type is used to infer if atom is link and if it is Ordered/directed or Unordered/undirected



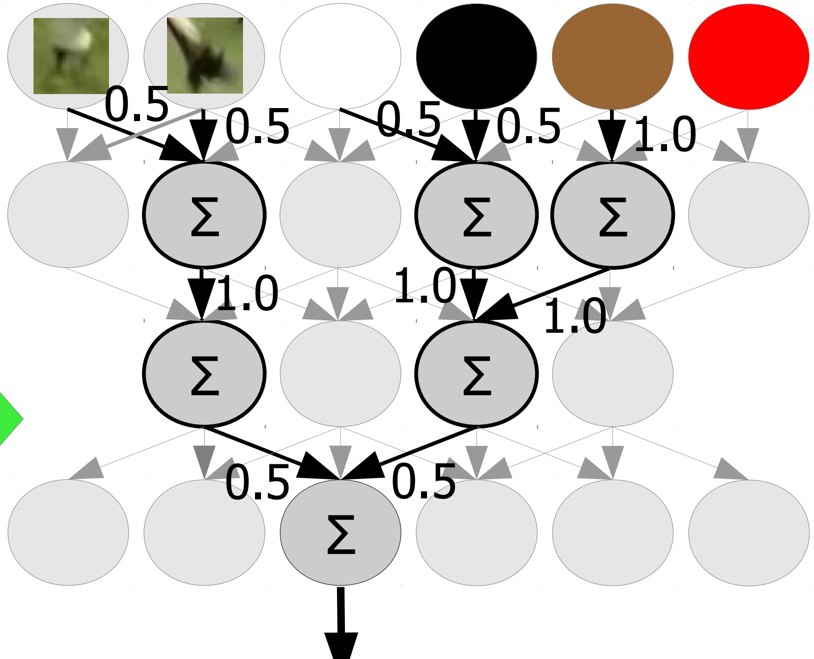
https://github.com/opencog/atomspace/blob/master/opencog/atoms/base/atom_types.script

Probabilistic Logic on Hyper-Meta-Graphs Making Unreasonable to be Reasonable



Explain

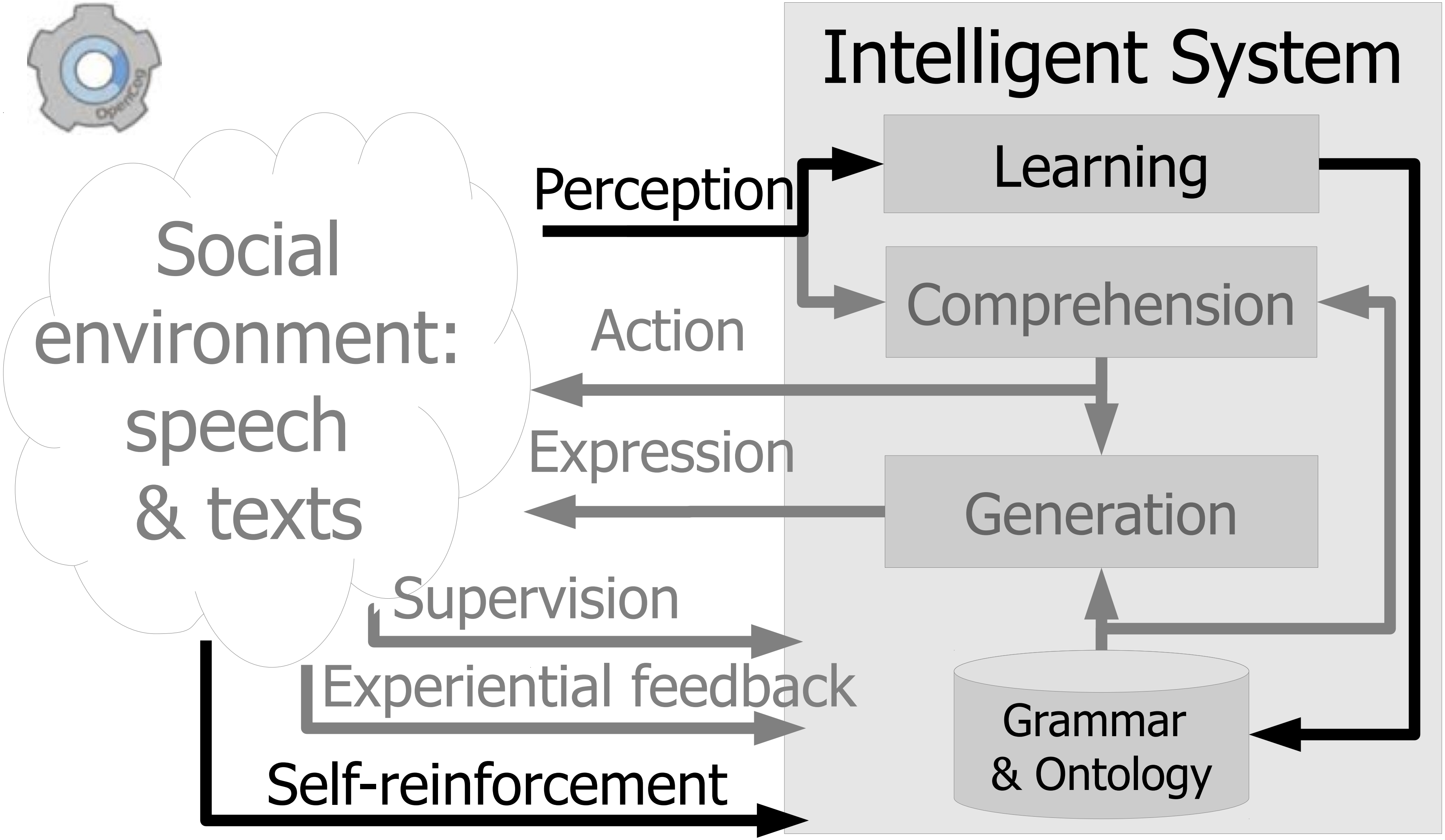
Transfer



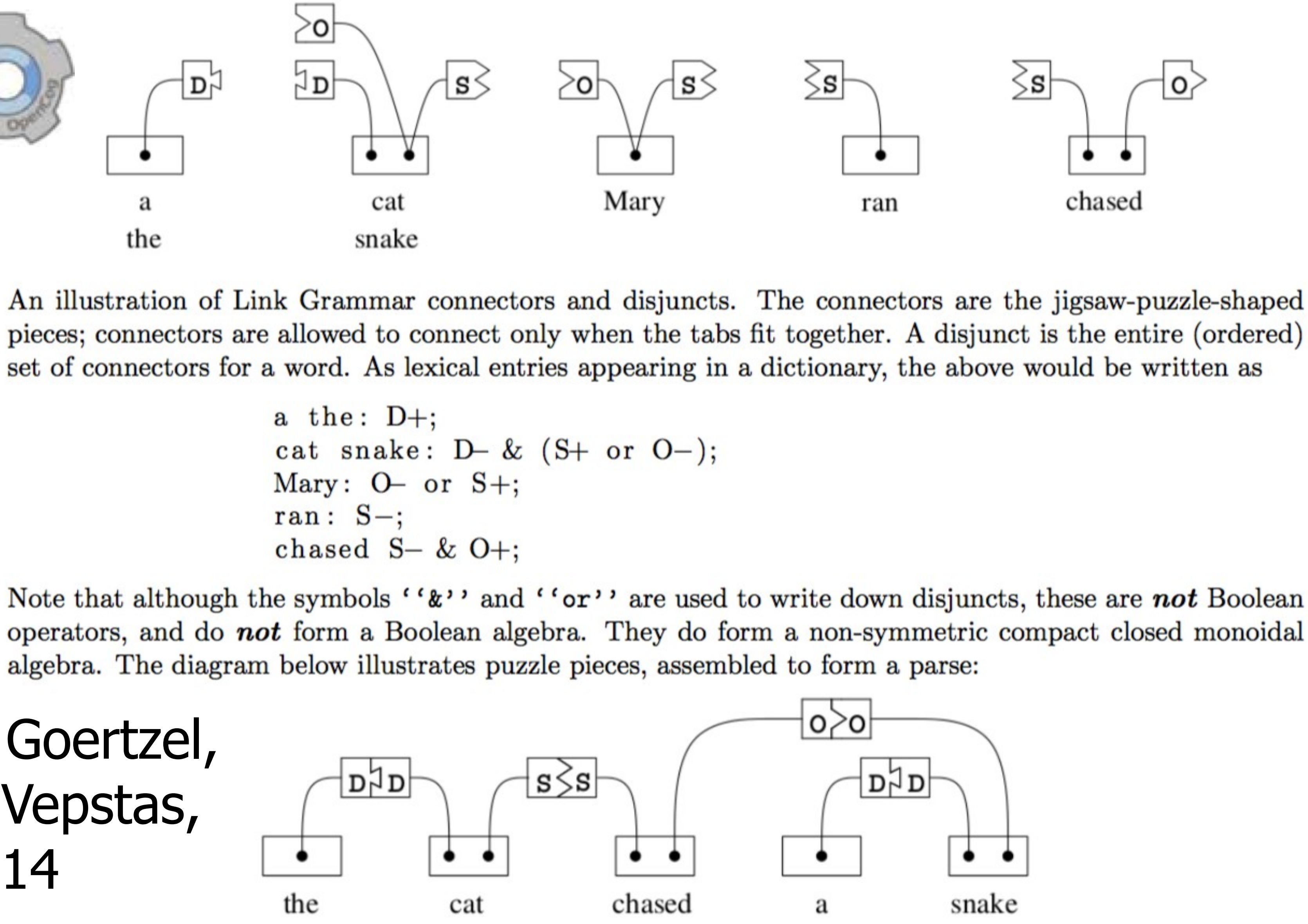
(Hooves AND Tail) AND
((White and Black) OR Brown)

=> Horse

Structured Language Learning Challenge



OpenCog Natural Language Graphs (Link Grammar – Disjuncts & Connectors)



An illustration of Link Grammar connectors and disjuncts. The connectors are the jigsaw-puzzle-shaped pieces; connectors are allowed to connect only when the tabs fit together. A disjunct is the entire (ordered) set of connectors for a word. As lexical entries appearing in a dictionary, the above would be written as

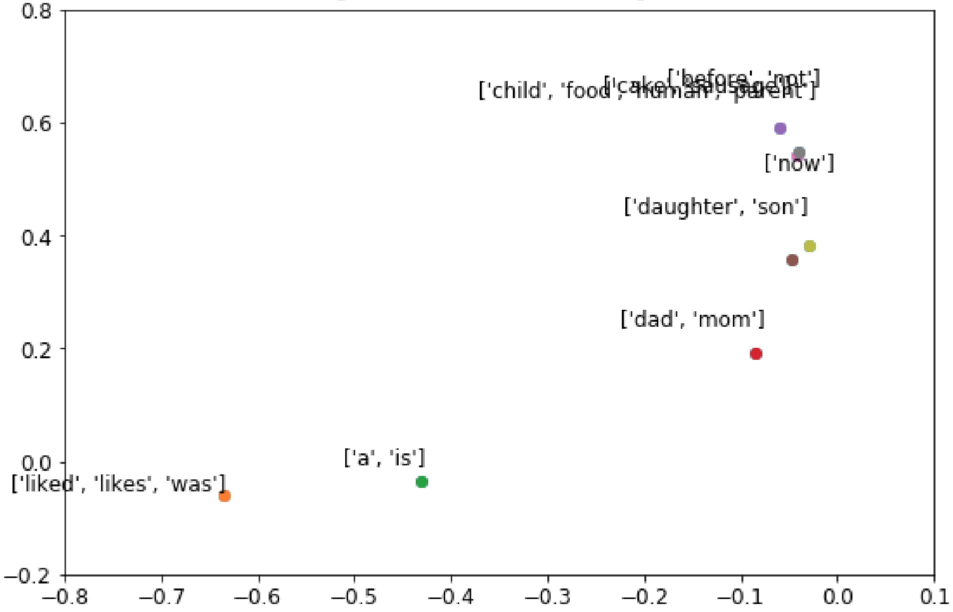
```
a the: D+;
cat snake: D– & (S+ or O–);
Mary: O– or S+;
ran: S–;
chased S– & O+;
```

Note that although the symbols ``&'' and ``or'' are used to write down disjuncts, these are ***not*** Boolean operators, and do ***not*** form a Boolean algebra. They do form a non-symmetric compact closed monoidal algebra. The diagram below illustrates puzzle pieces, assembled to form a parse:

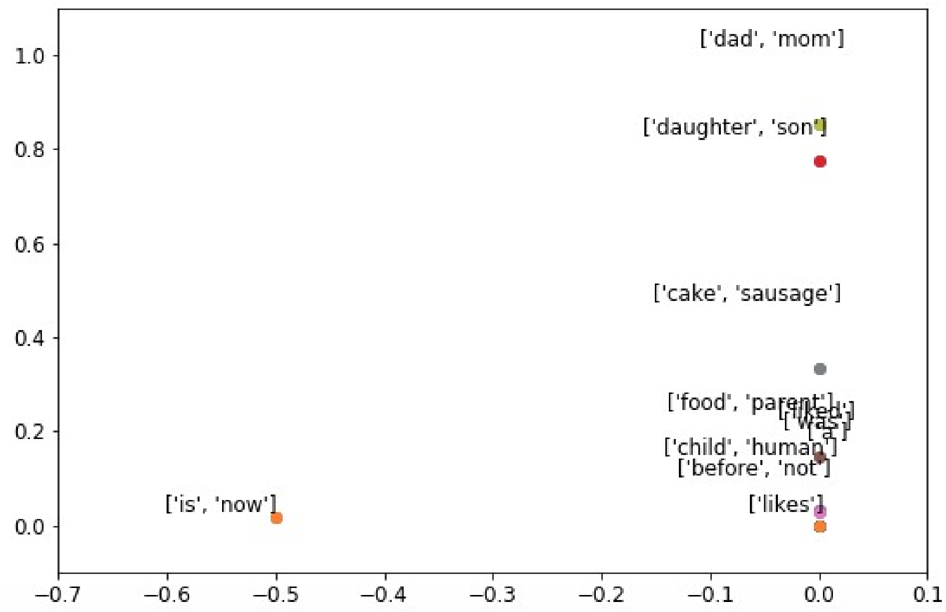
B. Goertzel, L. Vepstas, 2014

OpenCog Unsupervised Language Learning of Grammatical Categories and Link Grammar Dictonaries

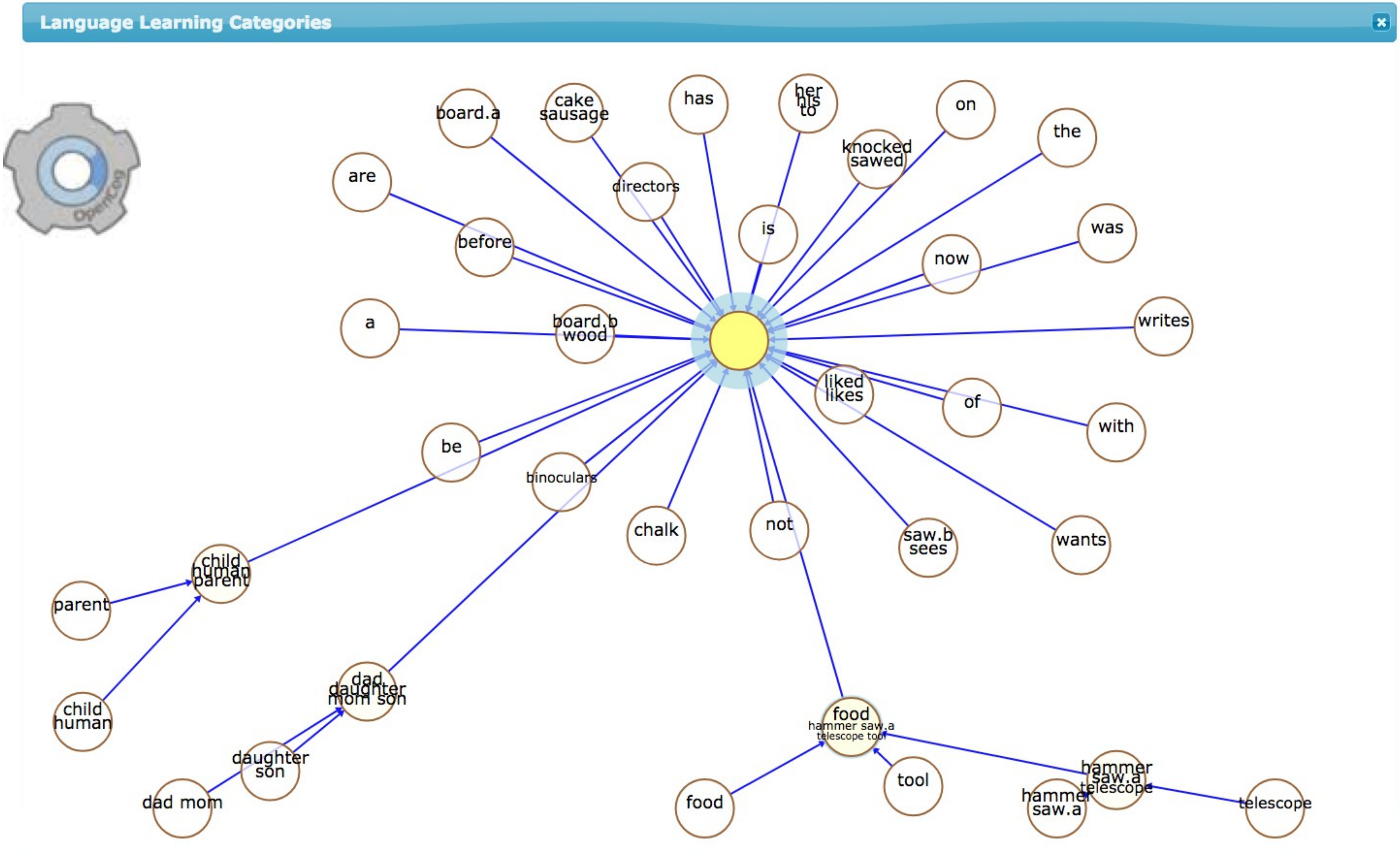
POC-English (Connectors)



POC-English (Disjuncts)



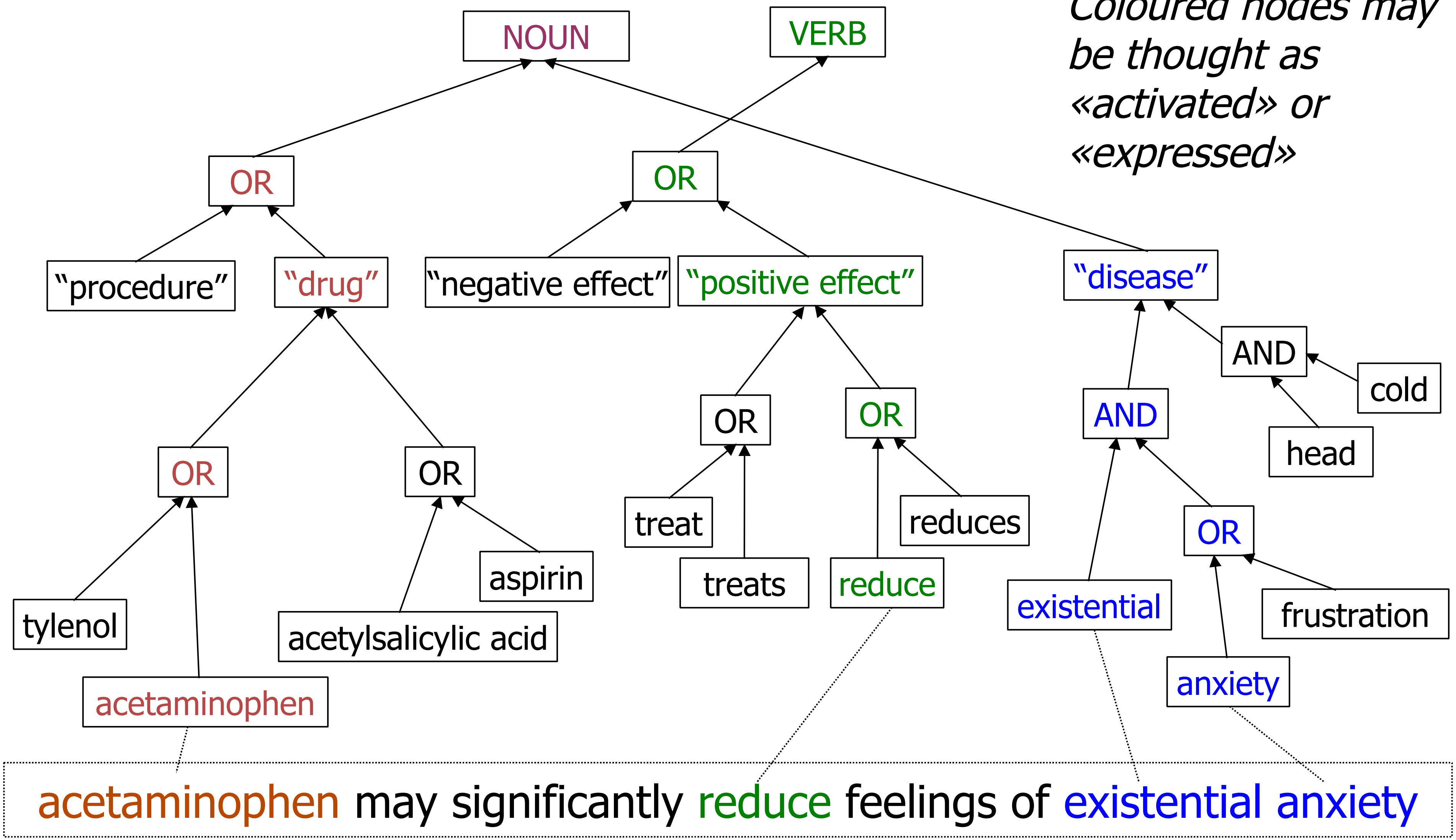
OpenCog Unsupervised Language Learning for Grammatical and Semantic Categories



Copyright © 2019 Anton Kolonin, SingularityNET, Aigents®

Aigents® Generic (Text) Patterns

Coloured nodes may be thought as «activated» or «expressed»



Aigents® Generic (Text) Patterns

```
<pattern> := <token> | <regexp> | <variable> | <set>
<set> := <conjunctive-set> | <N-gram> | <disjunctive-set>
<disjunctive-set> := { <pattern> * }
<conjunctive-set> := ( <pattern> * )
<N-gram> := [ <pattern> * ]
```

Example:

{[$description catheter] [$coating coating] [$inner-diameter {diameter inner-diameter}] [$tip tip] [$pattern pattern]}

X

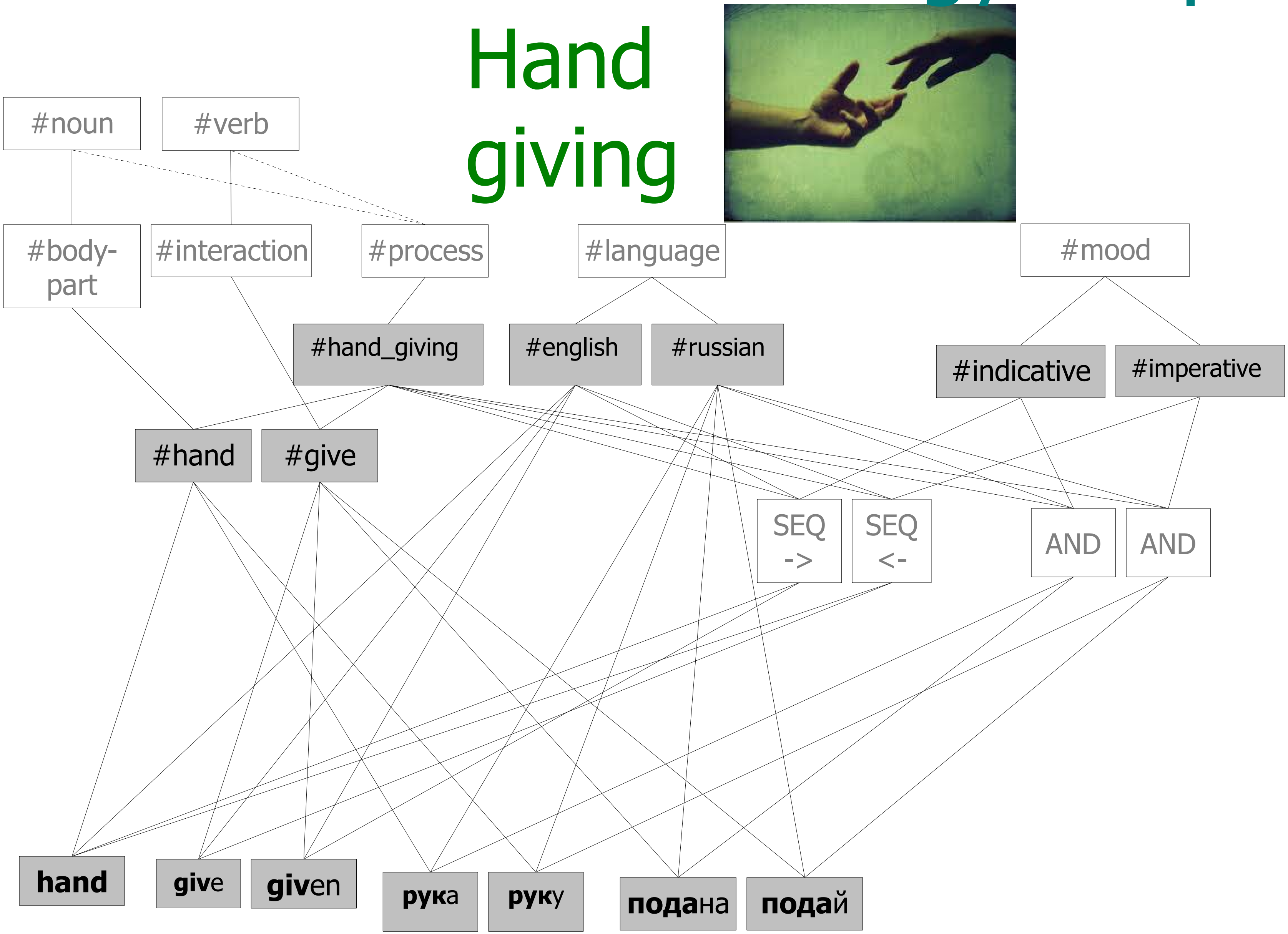
Convey Guiding Catheter. Unique hydrophilic coating. Small atraumatic soft tip. Ultra-thin 1 × 2 flat wire braid pattern

=

{ coating : 'hydrophillic', description : 'convey guiding', pattern : 'ultra-thin 1 × 2 flat wire braid', tip : ‘sof’t }

Grammar & Ontology Graph

Hand giving



#noun

#verb

#body-part

#interaction

#process

#language

#mood

#hand_giving

#english

#russian

#indicative

#imperative

#hand

#give

SEQ ->

SEQ <-

AND

AND

hand

give

given

рука

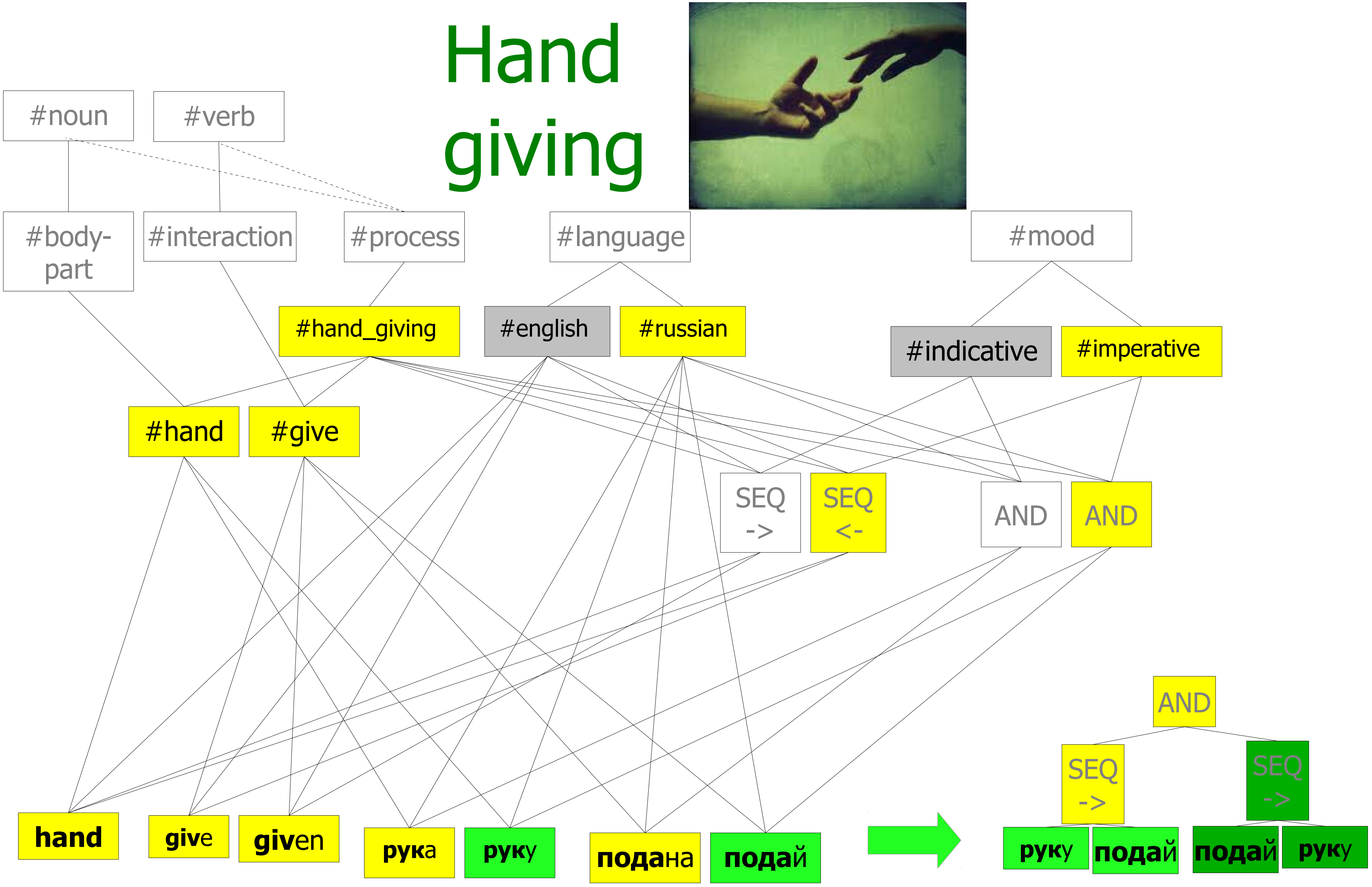
руку

подана

подай

Grammar & Ontology Graph Expression

Hand giving



#noun

#verb

#body-part

#interaction

#process

#language

#mood

#hand_giving

#english

#russian

#indicative

#imperative

#hand

#give

SEQ ->

SEQ <-

AND

AND

AND

SEQ ->

SEQ ->

hand

give

given

рука

руку

подана

подай

руку

подай

подай

руку

Copyright © 2019 Anton Kolonin, SingularityNET, Aigents®

Challenge – Integration of Syntactic (tokens and “word-pieces”) and Semantic (“Knowledge Graphs”) Representations for Context-based Word Sense Disambiguation

Какой (свойство зрения)?
Какой (состояние опьянения)?
Кто (профессия)?
Кто (имя, кличка)?
С чем?
Чем?
Что делал?
Где?
Как?

Косой косой косарь Косой с косой косой косил на косе косо.

Drunk oblique mower Kosoy with a slanting spit was mowing on a bar obliquely.

Thank you for attention!
Questions?

Anton Kolonin
anton@singularitynet.io

Novosibirsk State University
*THE REAL SCIENCE

AIGENTS®
https://aigents.com

SingularityNET
https://singularitynet.io