A Spacetime Approach to Generalized Cognitive Reasoning in Multi-scale Learning

Semantics and knowledge representation from observation, measurement, and learning.

Mark Burgess

August 2016, revised Jan/Feb 2017

***Abstract*—In modern machine learning, pattern recognition replaces realtime semantic reasoning. The mapping from input to output is learned with fixed semantics by training outcomes deliberately. This is an expensive and static approach which depends heavily on the availability of a very particular kind of prior training data to make inferences in a single step. Conventional semantic network approaches, on the other hand, base multi-step reasoning on modal logics and handcrafted ontologies, which are *ad hoc*, expensive to construct, and fragile to inconsistency. Both approaches may be enhanced by a hybrid approach, which completely separates reasoning from pattern recognition. In this report, a quasi-linguistic approach to knowledge representation is discussed, motivated by spacetime structure. Tokenized patterns from diverse sources are integrated to build a lightly constrained and approximately scale-free network. This is then be parsed with very simple recursive algorithms to generate 'brainstorming' sets of reasoned knowledge.**

I. Introduction

Reasoning has long been associated with formal logic in computer science, but it may be argued that reasoning is only a subset of a wider class of narratives, which may be told by joining together assertions. Logic's goal is to maximize the certainty of a narrative derived from prior concepts, by transforming them according to a constrained and largely deterministic set of rules, in which each step selects a unique possibility. However, it is only of value in a very limited set of circumstances. In other cases, where unique certainty is not available or practical, a more expansive kind of 'brainstorming' is needed for problem solving: one that allows multiple possibilities to remain open for selection at a later time.

Brainstorming is the first stage of a process of reasoning by 'whittling'. The idea is to create a large hypothesis set, for subsequent reduction into something more focused. The whittling is often emergent and iterative. Reasoning, in this interpretation, is a cognitive process, which allows an observer to integrate experiences, perhaps not evidentially linked, or even provably true, in order to form a set of speculative hypotheses. By process of contextual elimination, this converges, iteratively, to a much smaller set, or a final answer, using a variety of criteria from freshness to relevance and importance. In this dynamical approach, criteria like semantic and dynamic *stability* of the stories are now more important than logical idealizations like 'truth' or ad hoc determinations of 'correctness' [1].

Narration, or storytelling, is an underestimated aspect of intelligent behaviour [2]–[4]. It is how learning individuals explain things in terms of prior experience, and make sense of the world. To form a story, we have to know the difference between correlation and causation: correlations do not have direction, i.e. no arrow of progression, which may be used to accumulate narrative storyline. Hence, correlation can only offer short-range explanations, with uncertainty that grows with the number of claims for similarity. Combining directed associations into a reasoned argument is a different problem altogether, one that requires the propagation of semantics from step to step.

How we combine concepts into stories for small sets is one thing; applying expansive reasoning to systems with vast numbers of sensors, sporting very different characteristics, is a whole new challenge. Imagine a scenario like the monitoring of a massive array of smart connected systems, e.g. an Internet of Things, smart buildings and cities, etc. Data of very different kinds are generated at many different scales, and from a vast number of different sources. How could we begin to interpret phenomena across different scales? What concepts do we need? The challenge of making sense of such data, at scale, urgently calls for a broader view of artificial reasoning, based on improved semantic labelling of collected data. This is how we might scale the meaningful interpretation of data. But it is not simply a case of collecting more and more data. The monitoring of sensory data would be incomplete without the ability to reason about observations. Reasoning requires the definition of a set of concepts, tied to operational goals.

This work summarizes a model of invariant storytelling, based on spacetime inference [5]–[7]. The approach reduces the computational complexity for story inference by orders of magnitude. It describes how semantic associations may be collected from arbitrary sensors and inputs, and how a knowledge representation can learn context-dependent interpretations of those inputs, in order to later generate explanatory narrative automatically. It summarizes the practical

and implementational aspects of the promise theoretic notion of 'semantic spacetime', and implements a number of experiments based on the model.

Promise theory [8] motivates a graph theoretical approach to semantic scaling [6], which is based on the observation that cognitive relationships ultimately derive from elementary spacetime relationships. A set of recursive structures leads to a partially deterministic set of connected paths, each of which represents reasoned 'stories', with explanatory content. In this work, I show how to apply these structures, tentatively, to simple cases.

II. The semantic spacetime (promise) model

Promise theory frames elementary questions about intent: how it scales by cooperative interaction into extensive spatial networks, and how its properties may express spacetime-like variations to mirror a representation of environment. Based on the considerations in [7], we expect a number of information processing stages in a cognitive learning system (figure 1):

1) A sensory apparatus for inputting exterior information.
2) A stage that replaces sensory patterns with semantic tokens, i.e. *naming*.
3) A stage that associates semantics i.e. meaning to the named tokens, by associating them as clusters called *concepts*, associated in predictable and *qualifying* ways.
4) An introspective story-generating stage, which is able to 'think' about or activate interior concepts, and feed the resulting stream of consciousness back into the post-tokenization stages (3 onwards), along side sensory inputs.

Although we are not used to thinking of systems (physical or software) explained as narratives, any organized arrangement may be understood in terms of stories, so our goal is completely general. Most discussions of so-called Artificial Intelligence (AI) discuss stages 1 and 2, this work deals mainly with stages 3 and onwards, which I believe are orthogonal. These latter stages have much in common with linguistics [7]; indeed, once one can map invariant patterns into a finite alphabet of tokens, every reasoning problem maps to a linguistic problem.

A simple prototype system, to explore the underlying principles, has been reconstructed from the earlier cognitive approaches applied to pervasive computing management in CFEngine [9], [10], and further modernized and extended in the Cellibrium project [11].

III. Separation of learning scales

Data from only a single sensory episode, localized in time or space, are of limited value, and do not typically lead to claims of deep knowledge. It is by revisiting experiences, i.e. by the iterative process of observing and learning that we build trust in a stable 'invariant' representation of knowledge (see figure 2).

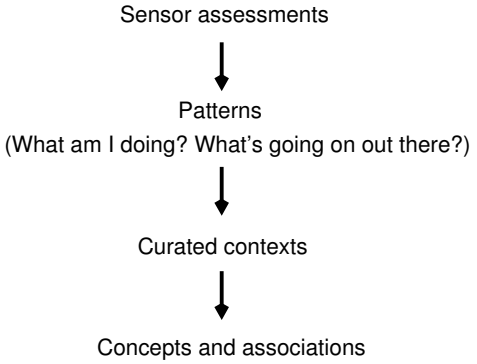

Fig. 1: From fast to slow, concepts are aggregated from co-activations over many scales. At each scale, meaning comes from the way we assign names to the aggregates.

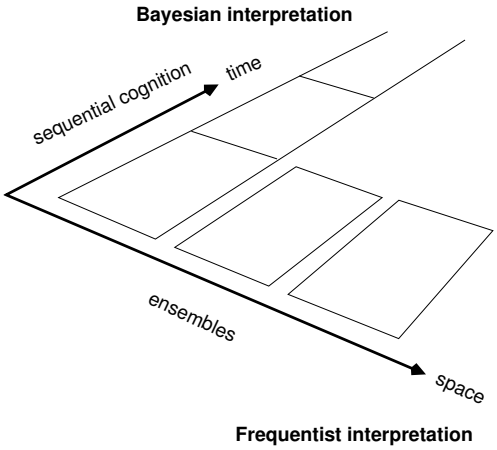

Fig. 2: We coarse grain space and time into regions of partially indistinguishable equivalence, e.g. different experimental 'trials'. Concurrence (in the same temporal grain), and coincidence (same spatial grain), define the boundaries for our cognitive experience. In cognition, every new time-like frame is a new experiment, and we must use learning to even out experimental inequivalence. Our (un)certainty is related to our choice of scaling or granularity.

Learning happens over a hierarchy of timescales [7]. Fundamental concepts are evolved by stabilizing 'genetic' adaptations through long term selection in a group; then there are concepts that build on this stability and use them to frame newly learned concepts, which congeal over generations, and are adapted and passed on socially (we may call this domain knowledge). This includes specialized sensors that tokenize, or dimensionally reduce, information intensive data into compressed conceptual representations [7]. Finally, there are concepts, which build on the foregoing, that are formed over shorter timescales by the observation and introspection of a single observer (see later figure 16). The latter is what we normally think of as learning; however, it is important to remember that it builds on a stable set of preconditions that have evolved prior to the current learning episode. We would expect consistent knowledge to be represented by eigenstates of a memory network.

In approximate order of aggregation, learning may be viewed at these scales:

- *Slow (evolutionary) training* (unsupervised adaptive learning): Random variations are whittled away into a set of behaviours and memories encoded in a long-term memory, and providing the seeds and the framing into which newer ideas are be formed through realtime recombination. The hypothesis here is that such ideas are naturally associated with the the most invariant aspects of space and time, leading to four basic kinds of conceptual association, as well as the basic ability to compress extensive spacetime datasets into simple tokenized concepts [7].
  In a software system, this is represented by hardcoded semantics, and specialized sensors, designed to discriminate pre-understood concepts, by virtue of their adaptation (e.g. sensors for hot/cold, light/dark, movement/no movement, face/no face).

- *Fast (direct) cognitive assessment or context* (unsupervised learning and recall): Context changes quickly, and some concepts and associations are formed at this timescale. Each independent observer inherits the adaptations learned by previous generations and can build on this basis, assembling context from sensory experiences, and assessing as emotional state by pattern recognition. The sensory channel will eventually be supplemented by an introspective channel (see last item). The effect of being simultaneously in mind, by observation or thought, leads to aggregate clusters, or composite concepts, by co-activation, which can be then named as new concepts.
  In software, this represents monitoring and classification of data. It might be represented by facial or handwriting recognition algorithms for training, etc.

- *Slow (indirect) training* (supervised linguistic learning): Given a lexicon of tokens (i.e. a language, however primitive), concepts inherited in a compressed form may be passed on from one individual to another, e.g. by word of mouth, or as text in a book. There is no longer need for direct experience. A set of concepts may thus be remembered in a knowledge bank, such as generational or societal memory, and be handed down as domain expertise. This form of knowledge acts as a second level of boundary conditions for framing and seeding new concepts. The fast cognitive knowledge is useless without having these seeds and constraints provide basic conceptual anchors. In life, the analogy would be the existence of domain knowledge that frames our current awareness of a situation.

- *Fast (recurrent) introspection* (emergent adaptation): Once a knowledge representation contains a sufficiency of concepts and memory representations to be able to form stories without new outside stimulus, a system can think about them and associate freely, leading to further new concepts. This would be essentially talking to oneself, as it would naturally share the same linguistic representation as the exterior sharing.

At each stage, the key question is how concepts are addressed and recalled, based on a mixture exterior and interior stimulus. This is what we mean by the naming of concepts [5][1]. This is discussed at length in [7].

Finally, no memory system would be complete without processes of annealing and garbage collection, to even out and clear away clusters and memories that do not become so important that they dominate over older ones.

IV. Encoding and retrieving conceptual associations

Data are singular events, with no a priori inferable interpretation. The meaning of a single data point can only be promised by its source. A knowledge system has to be able to integrate the diversity of such experiences into a stable aggregation, channelling perceived conflicts into contextualized differences, thus preserving them without neutralizing them. In a sense, diversity is a prerequisite for knowledge, and integration leads to a scaled form of hashing.

A. Direct primary cognition

Cognition leads to an association of tokenized concepts through a process illustrated schematically in figure 3. This prism separates data into a spectrum of four basic associative types, known as the irreducible types (see table I). It mirrors ideas in linguistics on the classification of nouns. [12]. The semantics of cognition are very different from the semantics of experimental observation in science. Ensemble measurement is about eliminating 'self' from experience. In cognition, observer subjectivity is a key aspect that cannot be disregarded[2].

B. Indirect secondary cognition (empathy)

In machine learning, one tries to fit bulk recorded experience, in all its detail, directly into a singular neural network, simulating direct first-hand experience by brute force reconstruction, then adapting

[1] This suggests that language would co-evolve with a conceptual representation of knowledge, and that utterances, i.e. statements and stories would grow in sophistication along side the knowledge evolved in a society of such representations or brains.

[2] A lot of attention has been given over to the algebraic aspects of measurement in physics, e.g. quantum mechanics, but surprisingly little attention has been given to the semantics of data, and cognitive perception, except in the case of experimental error [13]–[15]. The extent to which measurement disturbs the system during the act of measurement affects what can be promised about a measurement. Measurables may have the property of 'compatibility' [16], meaning that measurement of one does not influence the measurement of the other.

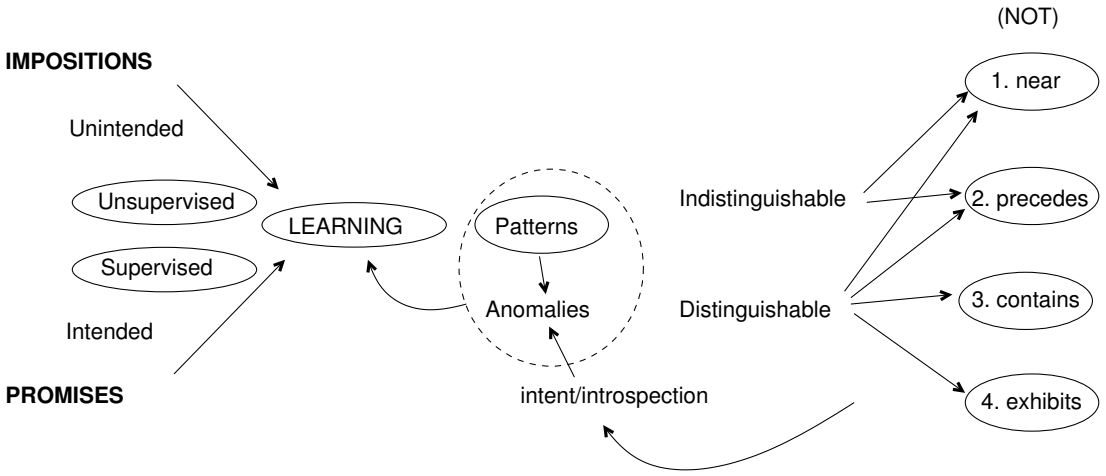

Fig. 3: A concept prism from left to right for association types.

its hardwired nature to match presumed semantics. However, humans are also able to learn from books and stories passed on verbally: we can learn from simplified, tokenized representations of knowledge, in which we do not have to experience every sensation and emotion as the source did. Even when learning to read books, neural network techniques would need to read every book into a network in order to train it, rather than using a short summary of what each book was about. As humans, we are able to scale knowledge acquisition through summarization, on trust, without verifying directly from a source. In order to scale machine learning, machines must also be able to do this.

When training a network by second hand teaching alone (e.g. from books rather than from practice or experience), we lose realistic sensory or emotional context, which forms part of our access key to recover the memories. Thus a model cannot simulate reality from a memory standpoint. Some aspects of context can be simulated or described second-hand, but a realistic emotional state will not normally match actual first-hand experience. The ability of a second-hand agent to *empathize* with the concepts and sensations of a primary agent will therefore play a major role in the ability of pass on knowledge[3]. On the other hand, the compression of such a complex lookup key, into a more compact address, allows to reorder concepts into new categories, and to generalize them, overlooking irrelevances based on context.

Cognition therefore ultimately leads to associations between tokenized (or dimensionally reduced) concept representations, through a whittling process, described in figure 3. This 'prism' separates data into a spectrum of associative types, which theory predicts to represent different spacetime characteristics, called the irreducible types [7] (see figure I). The conceptual prism bears some similarity with the staged structures of neural networks; however, neural networks are relatively expensive compression algorithms, that are not well suited to the latter 'expansive' stages of reasoning, where the bulk of input data would make reasoning too slow.

[3]This introspective simulation of emotional context may well be the evolutionary purpose of empathy, given the enormous value of saving in communication and computational processing.

C. Tuple form of semantic graph

Let us assume that the result of a learning process is a set of tuples of the form:

$$(C_1, \tau, A_+, C_2, A_-, \chi^*) \tag{1}$$

where C_1, C_2 are dimensionally reduced concept tokens, A_+, A_- are forward and backward associations of type $\tau \in \{1, 2, 3, 4\}$ and $-\tau$ respectively, and χ^* context labels. These tuples represent links or edges in a graph $\Gamma(C, A)$ whose nodes C are conceptual tokens, and whose edges A are associations. This summarizes an associative relationship implied by an observation. We can also add a weight or relative strength for each tuple and a timestamp for when it was last updated (which in a full knowledge equilibrium, with garbage collection, would be equivalent). All of these elements are strings. For example, highlighting the main features of the tuple, we may represent a simple qualitative description as a tuple:

doctor	concept
4	ST type
promises	forward association
identity credentials	next-concept
is promised by	reverse association
patient doctor registration	context

In other words, we know that a doctor promises to have identity credentials, and we are thinking about patient-doctor registration. This association is of type 4, on the prism. We can apply the same approach to turn numerical data into qualitative linguistic representations in this format, so

$$X = 47 \tag{2}$$

might translate into:

X,	concept
4	ST type
has value,	forward association
identity credentials	next-concept
is the value of by,	reverse association
variable assignment	context

In other words, the token X has a value of 37, and we are thinking about variable assignment.

What is noteworthy about the tuples is the absence of data types for the tokens, translating in an untyped graph. This is not a conventional data representation; rather it is a symbolic (purely linguistic) representation. This must be so, since semantics can only be extended by reference to other concepts, and these must be rooted somewhere in language. All concepts are thus symbolic strings, and their interpretation lies in the way they are associated.

Given such a tuple, nodes and edges are created for C_1, C_2, χ^* and a root node $C_{\rm all}$ directed associative edges are added for:

$$C_1 \xrightarrow{A_+, \chi^*} C_2 \tag{3}$$
$$C_2 \xrightarrow{A_-, -\chi^*} C_1 \tag{4}$$
$$C_1 \xrightarrow{ST\ 3} \chi^* \tag{5}$$
$$C_2 \xrightarrow{ST\ 3} \chi^* \tag{6}$$
$$\chi^* \xrightarrow{ST\ -3} C_1 \tag{7}$$
$$\chi^* \xrightarrow{ST\ -3} C_2 \tag{8}$$
$$\chi^* \xrightarrow{ST\ 3} C_{\rm all} \tag{9}$$
$$\tag{10}$$

With this linkage, we can determine which associations A_i are relevant to which contexts χ_j and also which concepts C_k may be activated by contexts χ_j. The encoding of concepts in this way, linked by associations, forms a recursive structure, which is described below. This is not regular in the manner of a Cayley tree: the result is closer to a 'semantic small worlds' graph [17]–[19].

V. Naming of concepts and the structure of a name

Semantic interpretation is essentially about the naming of things, in a representation meaningful to the observer. A name is a pattern (behavioural or symbolic) that represents a concept. Names are most useful if they are shared between multiple agents, so that concepts can be communicated and committed to shared (societal) memory, and be used to explain (not merely recognize) phenomena. Thus, the way in which we name observed patterns is of vital importance to their interpretation by other agents. This includes numerical names such as coordinate tuples, e.g. $(1, 2, 3, 4)$, (x, y, z), etc.

Without a consistent pattern representation for concepts (which we may define to be a language) there could not be recall of memory, and thus concepts would be useless and irretrievable. Thus language is a necessary condition for semantic interpretation. In neural network approaches, the dimensional reduction of a number of inputs to a small number of outputs is only meaningful when one can clearly and unambiguously identify the output channels with named concepts. This too is a simple language transformation[4].

A. Associations and their aliases in context

As argued in [7], the fundamental basis for conceptual association has its origins in the structure of spacetime, represented by four types of relationship τ (being close in space and time, being in a bounded area, etc). We then give specific associative meaning $A(\tau)$ to these different spacetime coincidences by naming associations according to abstracted concepts. Concepts and associations are nodes and edges of a graph:

$$C_1 \xrightarrow{\mathcal{A}(\tau)|\chi} C_2 \tag{11}$$

where the edge of the graph depends on a fundamental spacetime type, whose qualitative description is given by one of the four irreducible associations documented in table I, and their negatives.

The conditional parameter χ represents a context set of terms, under which this edge is active. Since context changes, by definition, as fast as we can think, or the environment can be perceived, the elementary tokens associated with context must be expected to be as large in number as the number of invariant sensory perceptions of the observer, and the most rapidly changing symbols in the knowledge representation. For this reason, we could expect primary context characterizations to be hardwired, i.e. built into the functional nature of the agent, by specialized eyes and ears, face cells and place cells, and so on [20]. Secondary characteristics could be learned over time, provided they remain simple enough to be activated quickly. It is unlikely that context would be expressed by complex concepts. However, it could be that this is the role of emotions: sensations as complex as sensory perception, but without strong rational linkage [10].

Multiple aliases, or alternative interpretations, of the four spatial relationships are possible, indeed they are encouraged in effective communication for expressivity and qualification. For example, type 1 (proximity) may be interpreted as adjacency, approximately equal to, close to, next to, etc. Type 2 (linear sequential order) may represent time, or unidirectional ordering, causation, dependency, etc. Type 3 (containment) may represent membership in a group, generalization of a collection of concepts, location inside or outside a perimeter, etc. However, we should also be cautious that informal association of linguistic metaphor also leads to confusions about the appropriate classification of meaning under the

[4]Note that this suggests that language is not a rule-determined system, but a pattern-based one, i.e. that any rules we identify are post-hoc rather than generative.

ST TYPE	FORWARD	RECIPROCAL	SPACETIME STRUCTURE
ST 1	is close to approximates is connected to is adjacent to is correlated with	is close to is equivalent to is connected to is adjacent to is correlated with	contiguity PROXIMITY "near" Semantic symmetrization similarity
	FORWARD	RECIPROCAL	SPACETIME STRUCTURE
ST 2	depends on is caused by follows	enables causes precedes	ordering GRADIENT/DIRECTION "follows"
	FORWARD	RECIPROCAL	SPACETIME STRUCTURE
ST 3	contains surrounds generalizes	is a part of / occupies inside is an aspect of / exemplifies	boundary perimeter AGGREGATE / MEMBERSHIP "contains" / coarse graining
	FORWARD	RECIPROCAL	SPACETIME STRUCTURE
ST 4	has name or value characterizes represents/expresses promises	is the value of property is a property of is represented/expressed by	qualitative attribute DISTINGUISHABILITY "expresses" Asymmetrizer

TABLE I: Examples of the four irreducible association types, characterized by their spacetime origins, from [7].

irreducible types, as interpretation by metaphor is fluid in human language [21].

Associations must retain their specific interpretation on every link, but also the generic type, which affects the way we parse the structure and hence reason about the associations. Meaningful narrative is usually driven mainly by type 2 (causally ordered sequences of happenings or reasoned steps), embellished by type 4 discriminating properties and lateral reasoning of types 1 and 3.

B. Scaling of semantics

Semantically enhanced concepts may be derived from the aggregation of primitive concepts, by associative clustering. Each cluster may act as a new compound agent, through its representative hub (whose name represents the aggregation), by making new associative links. The hub acts as a gateway to the qualified concept's associative network, much as a router is a gateway between a subnet and the larger Internet. Each concept is therefore an agent, at some scale of aggregation, with a unique name representing its meaning. In addition, the names of contextualized associations, within a cluster, and without, add the specificity of concepts expressed.

Based on trial and error experience, it seems that the most comprehensible or 'natural' linguistic naming strategy is one that follows the scaling of agency, i.e. aggregation, as described in [6]. With a scaled approach, one combines the names of interior promises to yield a compound name for a collective concept, and exterior promises then express the associations between the composite concepts [6]. In linguistics this is called *compounding*. The scaling is compatible with patterns observed in cognitive linguistics [22], indicating that language, as we understand it, forms a good network structure for concept representation[5].

From a cognitive perspective, the way we turn data into concepts depends on the spacetime boundary conditions imposed by exterior environment. The semantics of data are different depending on whether samples originate from from a single source or from a collection (statistical ensemble) of sources. In information culture, people sometimes talk about *pets versus cattle*, meaning unique instances with names and labels, versus herds without distinct identities. In data terms, these are singletons and ensembles.

C. The role of approximation in scaling invariance

In a batch experiment, where multiple samples contribute to a collective impression, all data points within an ensemble are considered to be equivalent, and with homogeneous semantics. In other words, we choose (as a matter of policy) to overlook any differences, and map them all to the same name. If a dataset is a stream of scalar values q_i, then every q_i would be considered interchangeable. The indices i may be relabelled and the results will be preserved. The properties that are derived from the dataset must therefore be invariant under relabellings of the array index i. This is how *invariant representations* are formed. Such invariant representations are crucial to avoid an explosion of contextualization, which would be expensive and would render concepts non-reusable. In order to learn from the past, a cognitive agent must transform ephemeral samples into invariant concepts efficiently.

For each of the irreducible spacetime association types, there is an interpretation of the meaning of aggregation, which promotes invariance:

[5]It should probably be considered a tautology that linguistic structure and conceptual networking are complementary.

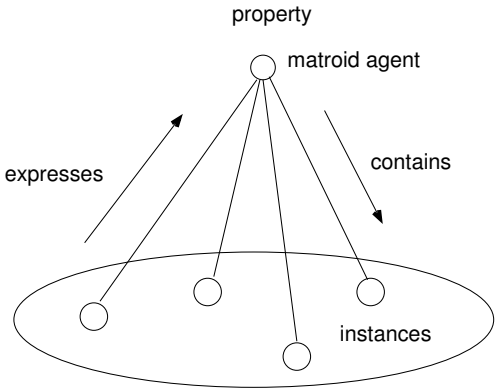

Fig. 4: The semantics of representing aggregation. If we group aggregated properties by a linking them centrally to a singular basis agent [5], [7], then attribute expression is effectively the inverse of set membership, and the basis agent alone represents the collective ensemble. Ad hoc clusters not have this property.

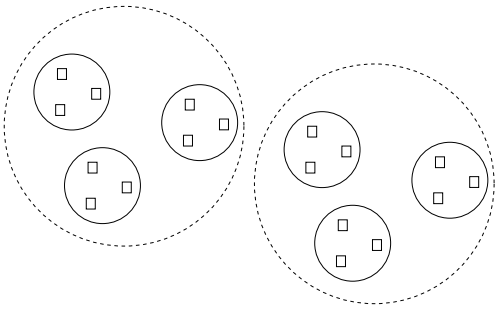

Fig. 5: Indistinguishable agents at different scales.

1) *Proximity clustering or approximation* (spacelike or semantic convergence): When named concepts are close together either in the real external world, or in the interior representation, then they form a labelled proximity cluster. The nature of the proximity may be physical, literal, or semantic depending on the label. The encoding is the same in all cases, up to a label. See figure 6.

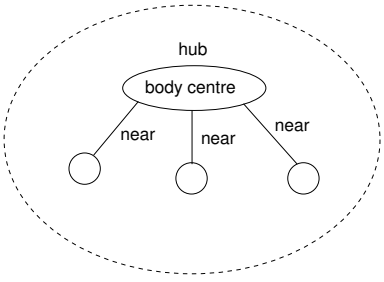

Fig. 6: ST Type 1 Proximity cluster is accretion.

2) *Causal aggregation* (timelike convergent dynamics): When dependencies come together to a head, so that a single outcome may be a causal determinant of multiple sources. This is a definition of the arrow of time. See figure 7.

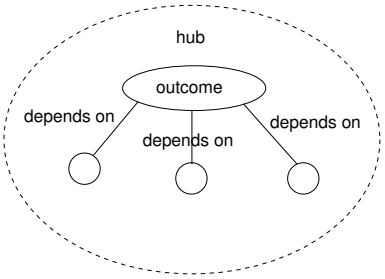

Fig. 7: ST Type (2) Causal cluster indication causation, like in a fault-dependency tree.

3) *Symmetry aggregation (membership or spacelike homogeneity)*: When concepts are joined to a single hub that represents their collective identity, by a 'contains/contained by' relationship. See figure 8.

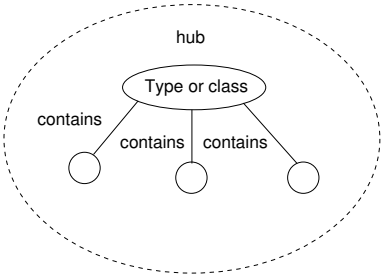

Fig. 8: ST Type (1) Symmetry cluster or membership class

4) *Qualification by attribution* (semantic qualification): When the name of a concept is qualified by a context (sometimes called a namespace), and possibly a role (like verb, noun etc), For example, the multiple interpretations of a word like "tar"
   - Tar as a noun in the context of waterproofing
   - Tar as a noun in the context of computing
   - Tar as a verb in the context of painting
   - Tar as a verb in the context of characterization

   must be distinguished using the hub construction for attribute association. This construction is a generic (pattern) form of what we call a schema or datatype in computing. It is recognizable without any specially arranged protocol, just from the observed associations. See figure 9.

The scaling of conceptual agency, in these representations, is a direct application of the promise theory [6], summarized here according to the four irreducible spacetime attributes elucidated in [7]. The parsing algorithm can now be be remarkably simple because the associations are organized around the most elementary

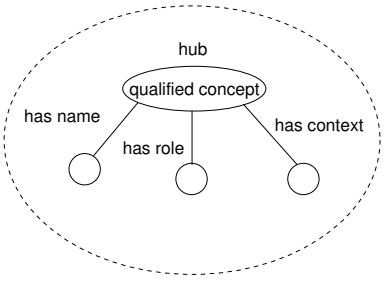

Fig. 9: ST Type (3) Semantic qualification clusters describe the kind of concept and its context.

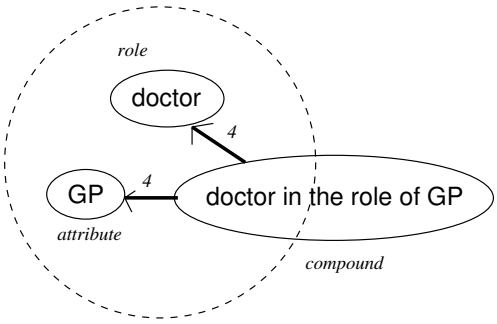

Fig. 10: Compound formation can be reduced to a functional pattern: a representative name linked to partial concepts (type -1 association) one of which is a main subject header (like doctor) This is a dimensional reduction strategy, because we can now refer only to the compound name and drag along all the conceptual baggage of its associations accordingly. Note, however, that the interpretation of a privileged role for is sometimes ambiguous.

spacetime concepts, or distinction in location, order, or expressed properties.

VI. The matroid naming construction

In earlier work [5], I showed how a promise view of structure motivated an identification of concepts as spanning sets, or matroids over a labelled graph. The most convenient representation of these is to bind all related clusters to a hub node, whose name is the collective name of the cluster[6]. This offers a scalable approach to spanning concepts [5], with upwardly extensible uniqueness. By associating a collection of associated concepts with a single point of address, we throw a kind of perimeter around the spokes and rim emanating from the hub to form a single new concept. Such clusters may overlap without limit, thus this is not a mutually exclusive branching process [3][7].

Unlike a hashing approach to unique naming, uniqueness can be assured by combinatoric compounding of names. The collection of named concepts in a cluster is accessed through a *single node hub* which acts as a gateway for the aggregation of parts. The name of such a matroid hub (or compound name) represents the name of the composite concept, formed by the internal association of interior concepts.

A. Nominal compounds and orphans

Concepts, whether primitive or derived, may therefore be thought of as 'lexical fragments', 'syntax fragments', or generally language fragment, formed by recursive scaling. A compound is, in essence, a long name, which contains evidence of the concepts it depends on, and which is linked recursively to the atomic concepts (see figure 11). It is also possible that the compound name forgets its association to the concepts from which it was derived, and survives independently through the frequency and associative density of its use.

[6]This is broadly analogous to the way we name a compound data schema object in computing. The hub has the name of the object in its entirety, and the members have the names of the associated items. The association types are analogous to the data types. Where the analogy breaks down is that matroid hubs can name arbitrary collections of data, they do not form mutually exclusive, non-overlapping memory segments.

[7]Concepts may be aggregated without limit for qualification, but they cannot be subdivided without limit. Thus taxonomies and ontologies as branching processes are not scalable to adaptation.

The study of compound terminology is a major topic in linguistics, which addresses the conceptual origins of names we use in language. For example, the leg of a table can be referred to by a single compound word or phrase 'tableleg', in which the compound has the role of 'leg' (by analogy to an animal leg) and a contextual attribute 'table' (see figures 11 and 10).

$$\Big(\text{leg} \xleftarrow{\text{role}} \text{(table-leg)} \xrightarrow{\text{qualifier}} \text{table}\Big) \qquad (12)$$

This example is straightforward, and fairly unambiguous. The name reflects the components directly with a conventional ordering, leading to a simple rule to construct a compound identifier. However, many compound concepts do not directly reflect their origins, or in fact forget their origins over time, e.g window (from Norwegian 'vind' and 'auge', meaning eye for the wind, clearly before the invention of glass). A window is neither a wind, nor an eye, yet the word serves its purpose with a whiff of the metaphysical. In other cases, there is no linguistic term for a new qualification, yet we need to distinguish ambiguous concepts with qualifying context, e.g. consider 'doctor'. A doctor may refer to a medical doctor, a general practitioner (GP), a surgeon, or someone with a PhD, etc. We could simply find unique names for all of these, but this approach does not easily scale; hence, it is normal to describe 'namespaces' as contextual constraints. This can be done by forming a compound name as a phrase 'doctor in the role of GP' for instance (see figure 10).

It is now somewhat ambiguous what is the role and what is the qualifier in this construction. It could be argued that 'GP' is the role and doctor is redundant, or that 'doctor' is the role and 'GP' is the qualifier. The distinction is somewhat arbitrary from a computational viewpoint (this is the danger of 'schema' thinking), but it is sometimes helpful to be able to distinguish a privileged component that describes the behaviour or function of the term. Through the various experiments conducted during the course of this work, from topic maps [23] to the present model, it seems that the

appropriate place for agent role in a network lies in the associations between pairs of concepts, rather than in the concepts themselves. If one tries to 'type' concepts (in a typology, taxonomy, ontology, meronomy (partonomoy), etc) one prevents the free association of ideas by analogy that is so central to human reasoning. Thus the classic idea of a 'data type' for concepts in a semantic network appears to be simply wrong because it leads to an over-constrained network with too much uniqueness. The resulting network is too sparse to allow percolation and hence propagation of storyline [7], [24].

Finally, there is the central problem: how do we think up names to concept clusters? The naming of clusters seems analogous to the matter of forming nominal compounds (see figure 11). One approach is to try to abandon linguistic meaning and use some form of unique hash. This approach is used in natural language processing [25] with some claimed success. However, hashing is a one-way transformation, which is unsuitable for our associative map. To create an interactive system of knowledge, we need to be able to read back concepts in a form parsable by humans.

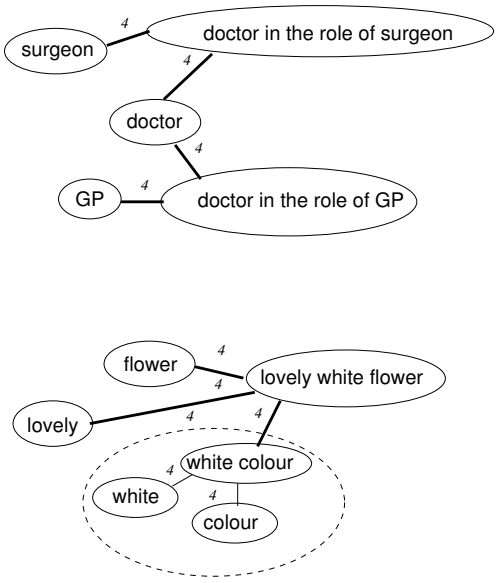

Fig. 11: Under aggregation by a hub, the irreducible types effectively collapse into a single type, bound by membership to the hub. This kind of central binding is the most invariant construct of all.

The matroid hubs, with their compound names, are binding points, or confluences of association act, which as semantic correlators, i.e. authoritative anchor points, expressing uniqueness. If we follow through this approach consistently, then conceptual specificity increases upwards with the level of aggregation, rather than downwards with the depth of branching, as in a taxonomic hierarchies (i.e. branching processes). This leads to a bottom-up approach to conceptualization, which is entirely compatible with the modern genetic understanding of phylogeny and phenotype expressed through the aggregation of genetic flavours. This is also motivated by a promise-oriented interpretation. Mirroring the proposed usage approach of language formation in cognitive linguistics [26]. It is a chemistry of semantic atoms, mixed into distinct molecular combinations, with differentiated functional meaning.

B. Concepts, associations and the scaling hierarchy

The end result of a cognitive process is to tokenize complex data inputs into tokens (concepts), which summarize the data in an *invariant representation*; this might be approximate, but it is proportionately immune to variations, and to link these together by association. In promise theory language, concept tokens are agents (nodes in a graph), and associations (graph edges) are promised by these agents.

There is a ladder from sensing to conceptualization, which proceeds by aggregation (over time and space). This involves work, and thus it takes increasing time to process complex associative concepts. Concepts are clusters of agents. The basis of interpretational semantics lies in the following observations:

Law 1 (Coincidence and concurrence): That which occurs together (concurrently, coincidentally, i.e. locally, or in the same context) is associated. Thus context belongs to associations, not to concepts. Measurements at different locations and at different times are not necessarily causally related, but may be correlated. The naming of the association is undetermined.

Law 2 (Semantic stability): Averages stabilize semantics by decoupling from temporal and spatial approximate regions. Data sampled and combined from different sources (locations) should be treated as ensemble averages, in a single concurrent experiment. The meaning of time and location are lost in these averages, and semantics are stabilized by this decoupling.

Law 3 (Scaled agency): Semantics (interpretations) arise from the naming of associated forms and patterns, in space and time, and across multiple scales. Names assigned become invariant concepts.

Law 4 (Constant semantics during observation): Measurements are comparable if they have the same semantics. In cognition, observations are not comparable in an experimental sense, unless we deliberately overlook certain information. Thus semantic stability depends on what information we throw away.

The last two are circularly self consistent.

C. The learned context channel

During activation, inputs might activate an elementary context token, and in turn, several such atoms might activate an association, somewhat in the manner of a crude neural network (see figure 3). The relative activation level of an association is thus the integral of the weights of its incoming edges. A triggering policy could also be employed for generality and tuning; however, we can only simulate this spatial interference process on von Neumann computers..

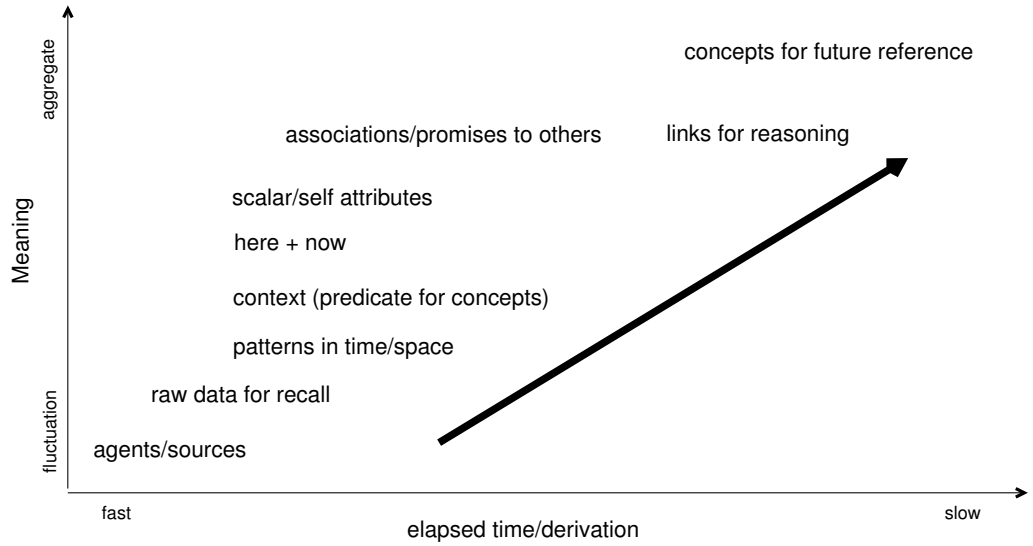

Fig. 12: Climbing the knowledge ladder takes longer and longer to advance towards sophisticated concepts, from fast sensing to pondering concepts, because of iteration and aggregation (learning).

Context is used to frame the circumstances under which associations and concepts are identified. Striking the balance between a context that strives for a coordinatized precision and one that is maximally invariant (for repeated 'context-free' use) is the central challenge of building an *addressing system* for semantics and knowledge [7]. In the implementation of Cellibrium, it is thus assumed that:

- Context is attached to associations, not concepts. A co-activation of an associative bond needs only be a set that activates both ends, but the specific members may differ at each end by evolution of contextual awareness of the cognitive agent.
- Limiting the amount of nominal information in context is a priority, else the memory requirements for knowledge would be divergent.
- All associations belong to one of the four spacetime association types [7]:
  1) An approximate similar location in space.
  2) A common cause or outcome, convergence of path in past or future.
  3) A common enclosure or membership.
  4) A shared property, leading to implicit membership (e.g. all kinds of doctor in figure 11).

  In aggregation by symmetry, all these collapse to type 3 in practice, i.e. grouping of indistinguishable members [7].
- Context describes something analogous to recent browsing history:
  1) The intent that led to the association being made (interior state).
  2) The emotional and rational state of the agent's recent thinking (interior).
  3) The unintended circumstances in which the agent finds itself when making the association (exterior state).

  In other words, anything we happen to be thinking about or involved in can be a context for knowledge relevance (see figure 13).

D. Interior and exterior associations of concept clusters

What is expressed externally by sensors plays a different role than what is expressed internally by introspection, as we saw in section VI. Interior and exterior properties continue to play a role in distinguishing. Figure 13 illustrates this. Both interior and exterior contexts may play a role in addressing, i.e. labelling and retrieving, concepts (see section VII-B).

There remains the question how we should use context to activate or deactivate story pathways. In an artificial system, especially one made for a domain specific purpose, the information content, or semantic complexity, inherent in context annotation during training might be too small to give a reasonable chance of discrimination of relevance in real-world circumstances. Unless there is a sufficient density of discriminators, we will either fail to find possible pathways (lateral thinking) or we will include too many, losing the point of context as a refinement criterion.

We might rank pathways by an activation score for how each node in a story overlaps with the current context. Alternatively, we might rank an entire story along its path. The latter would bias in favour of long stories[8].

[8]If we could find a normalized 'probability' to sum per unit story length, then we could build a score analogous to the Shannon entropy, in which associations were symbols in an associative alphabet. This remains for future investigation.

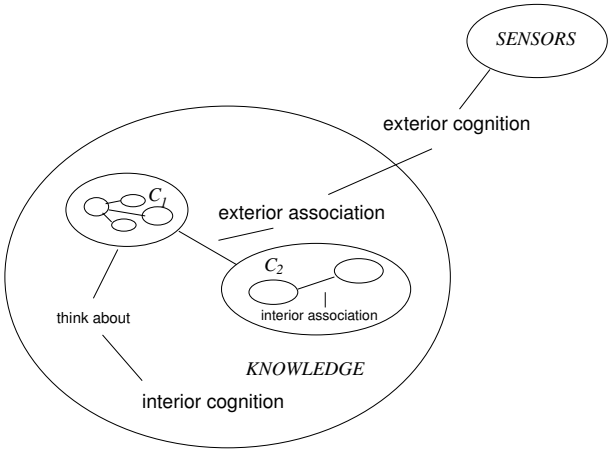

Fig. 13: The scaling of association and cognition. Context labelling of associations may be interior to compound concepts, or exterior between concepts. Context acquisition, by cognition, may be interior to an agent observer (intentional or introspective) or exterior (sensory or unintentional).

E. Examples of contextualization

The structure of the following examples helps to show why context is so important, and why it is important to aggregate clusters into concept-property matroids and situation-context groups. This is a set-theoretic interpretation. Contextual information is both interior (explanatory of subjects) and exterior (causal and situational or correlated). The following cases help to exemplify the issue:

- The doctor (in the role of surgeon) depends on surgical gloves in the context of (operating room,examining patients).
- Bruce Wayne (in the role of Batman) is a member of super-heroes (in the role of night-club).
- Tidal is generalized by (is a kind of) company when (thinking about music, streaming music)
- Tidal in the role of gravitational effect does NOT have the role of company.
- Tidal is NOT caused by the moon in the context of tidal as a company.
- Tidal has the role of an effect when thinking about physics, gravity.
- Gravity depends on mass in the context of physics.
- Gravity is the name of a book (a book called gravity) in the context of physics.
- Temperature expresses very hot in the context of measuring server.
- Hot contains very hot in context (all contexts).
- Very hot contains '50 degrees C' in the context of measuring server.
- Temperature expresses hot in the context of (measuring server, 306, datacentre, NYC).
- Temperature expresses hot in the context of June.

These singular context names, while meaningful to readers with a broad cultural knowledge, are too simple to allow meaningful discrimination during machine processing. We need to understand how to extend context labels for efficient addressing of semantic relevance.

F. Context design and compound concepts

Because of the recursive nature of concepts, the promise model of scaled agency in [6] suggests that it will be helpful to distinguish between *interior* and *exterior* context of a concept. Interior context refers to qualifying associations positioned 'behind' hubs, which help to support and qualify the interpretation of a concept. Exterior context refers to the set of concepts that are active in the recent train of thought of the total cognitive system, as it interfaces with the world on the other end of its sensory apparatus (see table II).

CONTEXTS	INTERIOR	EXTERIOR
LEARNING SOURCE / AWARENESS	Introspective thinking	Observational sensing
MEMORY ADDRESS / RETRIEVAL	Qualifying property	Relative placement

TABLE II: Semantic scaling leads to a notion of interior and exterior at many levels. These distinctions play an important role in defining cognitive linguistic recursion during the learning and retrieval of stories.

Interior memory context acts as a kind of 'type' or 'role' interpretation for concepts that have been selected or deselected by an exterior *memory context*, whereas interior *awareness context* represents the feedback of the system talking to itself about past experiences and thoughts. In all cases, the functional treatment of memory context may be handled by an invariant algorithm, with great computational simplicity. Context is not just about the addition of descriptive words during learning, but also about linking those compound states to all the sub-concepts on which they depend too. By following this approach, the average degree of connectivity in the graph increases *superlinearly*, making non-trivial stories increasingly likely, while also providing switches by which to exclude pathways based on later context during retrieval.

The recursion in the graph corresponds to a recursive linguistic structure that can be illustrated as follows. Consider the following unary association, containing subgraphs, labelled by their irreducible ST-type.

(the cat which (is black (4), eats fish (2)))
expresses*(4) (happy purr) in the context of (living room)*

```
// What kinds of compound contexts can we expect? State of mind...
// What are we trying to do?

ContextCluster("doctor service");
ContextCluster("patient appointment");
ContextCluster("patient health service");
ContextCluster("need to visit a doctor");
ContextCluster("patient doctor registration");
ContextCluster("identity authentication verification");

// Compound (qualified) concepts

RoleCluster("GP doctor","doctor", "general practitioner", "patient health service");
RoleCluster("surgeon doctor","doctor", "surgeon", "patient health service");
RoleCluster("patient appointment","appointment", "doctor patient", "patient health service");

// Elementary bidirectional associative links

Gr("doctor",a_promises,"doctor availability","patient doctor registration");
Gr("health service access",a_depends,"patient authenticated","need to visit a doctor");
Gr("health service access",a_depends,"patient appointment","need to visit a doctor");
....
```

Fig. 14: Example programming statements leading to the formation of a graph from within a programming environment. Any system could, in principle, be instrumented from within to offer semantic hints for aggregation by a central 'brain'. Helper functions divide up and generate patterns, as illustrated in figures 10 and 11.

The parenthetic recursions are all *interior properties*, and the explanations of situation are *exterior* (boldface). We end up with useful statements of the general structure:

(
(concept name, interior recursive properties)
(association type, alias)
(concept name, interior properties)
in (exterior context)
)

This suggests that an API, something like the one shown in figure 14, can help to automate the structural regularity during the documentation of concepts and associations, i.e. during semantic learning. Notice how, by using the matroid hubs and compound concepts to reduce the dimensionality of conceptual activations into a single concept, we also achieve an economy of scale, with attendant computational simplification[9].

VII. Retrieval of 'knowledge'

Once encoded, retrieval of associations is achieved by reparsing the representation in figure 3, subject to an independently maintained context. Sensory context is a set of somewhat elementary tokens, which activates certain concepts and associations. Exterior context may be exchanged for slower, but more stable introspection, in which derived concepts are fed back into the context by 'thinking about' a particular topic. Tokens can be followed individually or activated in parallel to search for the consequences of a particular concept in the current context. Implementing this context channel will be the most challenging aspect of building a cognitive knowledge representation. Only the first stages will be addressed in this report.

A. Conceptual pathways

Retrieval of explanations and associative reasoning can be open ended or bounded (see figure 15); we want to:

- Explain how to get from a starting concept and a final concept (bounded but not unique).
- Explore where we can go from some a starting concept (unbounded brainstorming).
- Explain the different ways one can get to a final concept (inverse brainstorming).

The first of these is somewhat analogous to a quantum path integral [1]. Any one or several of the possible pathways might be in play between concepts in a given context. For example, without sufficient context to know the interpretations of 'Oslo' and 'America', both these unary associations may be in play simultaneously during a reasoning process:

- Oslo is located in America
- Oslo is NOT located in America.

In logic, this would be considered impossible, yet both of these pathways may be equally relevant (we should avoid the expression 'true') if the context is unable to select only one. A knowledge system cannot even know that these seem potentially contradictory without a sufficient density of linkage to qualifying context. We must be quite careful not to over-constrain reasoning by projecting human expectations onto a process too early. It is likely that concepts such as true and false are emergent rather than prescriptive.

[9]It is interesting to speculate whether this observation might help to explain the common use of compound words and special terminology, like acronyms, in human language.

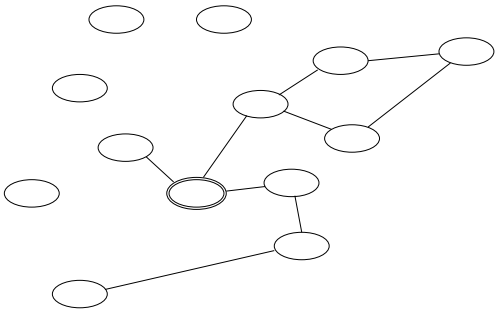

Fig. 15: Brainstorming around a start concept, without a clear end. Reasoning may contain loops, which need to be handled somehow.

B. Context in retrieval

As we trace along a story, from concept to concept, four semantic component sets may be associated with with each step in a story (see figure 16).

- The current concept in a story (a fixed boundary condition).
- The concepts associated with the current concept (directed links to possibilities).
- The current awareness context or 'state of mind' of the observer (current relevance).
- The awareness context at the time of learning (encoded in interior and exterior associations).

How these four sets activate one another is key to how stories may be generated 'deterministically'. The current concept is a natural part of the state of mind of an agent, and thus it belongs to the interior cognitive context as much as any sensory inputs. The cognitive context of an agent divides into interior and exterior parts. Interior context is what the agent is thinking about, independently of its sensory inputs. The exterior context is its sensory channels. Thus some context information is naturally inward-looking, or introspective, while other information is outward-looking (grounding in the realities at different times). It is interior awareness context that determines the *relevance* of forward pathways, because state of mind is the calibrator of intent. Exterior awareness context is about framing the current experience, grounding it by experience or feeling. Thus, the two cognitive 'awareness' channels have different semantic roles [7].

As context evolves, so does the focus of consciousness, or 'what the system is thinking about' i.e. the current concept or set of concepts in a story frame. Interior associative context plays the role of functional type, and is entirely encoded in the recursive compounding structures of the knowledge representation. So when we retrieve knowledge, this kind of context is purely qualifying or explanatory of the usage. The major links to concepts will be to the fully qualified hub nodes, and the interior context will be hidden unless explicitly examined, at any given level of recursion.

It is exterior associations that link concepts to one another in the knowledge representation, so they determine which concepts lead to other concepts. However, those exterior associations are judged for relevance by comparing the context in which they were learned with the current context or state of mind of the agent. This leads to a complex causation: concepts overlap through association, and context overlaps with pairs of concepts, and perhaps beyond. We therefore have two channels evolving in parallel: a short term memory of context and a long term memory of qualified concepts [7].

C. Evolution of awareness context

Measuring the relevance of forward pathways remains an unsolved problem. How should the current context evolve then as we traverse a story. How quickly should context be forgotten? When to increase the scope of context and when to say that our evolving thoughts have drifted off-topic is an highly subtle determination, which may not lend itself to algorithmic or pattern based solution. It might well be that relevance needs a bank of multiple pathways interfering across the entire length of a story path. When we apply software to the problem, it is natural to treat each leg of a story as a Markov process, each step independent of the last, with some activation context adding transverse memory. This is probably too simplistic, but for this work, that must be the limit of ambition.

As a first approximation, we may take the lexical overlap of the two context sets (past and present) as an estimation of relevance. Using this, we can sort stories link by link to judge the relevance of each twist and turn. The relevance of the total story cannot be calculated without a particular end-topic. That goes beyond the scope of these notes.

When machine learning is applied to document processing, story fragments are treated as patterns to be recognized, and the relevant forward pathways are determined based on statistical ensemble support rather than semantic relevance [27], [28].

There are many issues with trying to match relevance based on context in a small system. Present context may not match the context at the time of learning, because there are too few degrees of freedom for coupling contexts together. It may turn out that we have to rely on more universal invariant context markers like emotional states (fear, happiness, etc) to play a role in activation and selection. Although these are poorly understood, they may play a central role in joining together ideas that cannot easily be joined by explanations of semantics.

As usual, it is important to remember that semantics are secondary to what dynamics enable, thus causal propagation of meaning cannot be assumed across different timescales (see figure 16). We may expect the timescales of learning and retrieval to become separated eventually, with retrieval times much longer than training times, yet we must still recall and modify stories that were learnt previously. This gives

some clues about the encoding. In particular it tells us that all causal information needs to be encoded in exterior associations, not by relying on context to light up answers directly, like a primary database key.

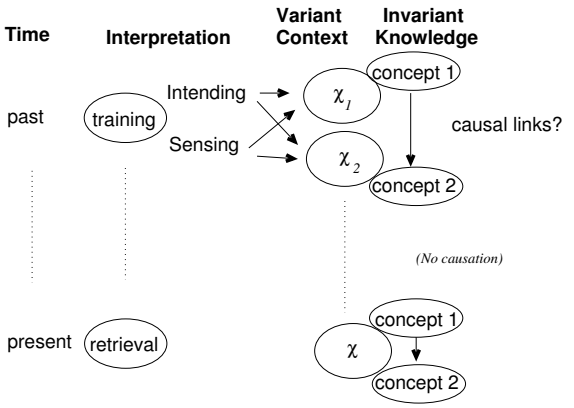

Fig. 16: The timescales associated with learning and retrieval of stories. The links between concepts may or may not be causal over long separations, but context is always causally related to concepts over short timescales.

If excessive specificity can be the enemy of retrieval, then specificity is important during the encoding phase of knowledge. For example, it is not the general concept of birds that depends on the concept of flight, but rather a fully qualified instance of a bird. In other words, we have to be careful not to confuse what is general with what is specific, at any level of recursion, as this can lead to manifest errors of reasoning, such as 'all birds depend on flight'. Matching of conceptual scales is yet another problem that must be deferred for further study in future work.

VIII. The experiments

At the time of writing, a simple proof of concept implementation of the foregoing cognitive system has been implemented. Only the simplest level of interior/exterior context conditioning was used to activate and screen paths, owing to the complexity of simulating such a feedback system in software. The initial goal was to illustrate the principles.

A. Data sources

Four kinds of data source were imported into a semantic graph, in the hope of first generating stories by brainstorming. Even this simple first step has application for causal analysis, qualitative hypothesis testing, etc. The data sources were:

- *Computer monitoring examples*: data from the Cellibrium CGNgine agent (a CFEngine 'computer immunology' derivative, with embedded machine learning at the pattern level) were output in realtime, as in a production datacentre environment or IoT scenario. As an intentional system, the intended policy could also be encoded and connected to the activities of the agent, describing desired state. Thus data observations, intentionally measured from the actual state of the system, and converted into invariant forms, with an intended interpretation relative determined by policy goals and constraints. See figure 18. Policy documentation files, describing intent, were incorporated, such as the intended meanings of software functions, with links to documentation, as well as the coding of particular policy rules. The source code of the software agent could be scanned to extract error messages given in a particular context[10]. Domain knowledge about the design and intent of the software itself was included manually, as well as remarks about the world of computer datacentres, servers, and operating systems.
- *Software system example*: Data scanned from the nesting structure and dependencies of software, packaged in process containers, file bundles, and linked binaries, probed using intended composition rules (source: Makefiles, container specifications, package compositions, and binary linker data from `ldd`, etc). See figures 19 and 20.
- *Doctor online registration wizard*: Domain knowledge from an imaginary online scenario, where a patient is trying to register for a doctors appointment in a new city. Concepts and associations are input directly from a wizard overview of knowledge of intent about the smart distributed service, and the software it uses, its dependencies, and workflows etc., (source: authors of the system). See figure 21.
- *Big picture semantic index example*: Domain knowledge contributed from possibly multiple sources giving a cognitive overview of an entire environment or experience. See figure 22.

B. Machine learning

Machine learning appears at two distinct levels:

1) The sensory data collectors sample the exterior sources at some (relatively frequent) timescale, in accordance with Nyquist's theorem. Each collector, with its independent semantics, transmutes sample data into a separate invariant representation and learns its significant patterns and states (for monitoring, see [29]). From this Bayesian statistical description, a lexical name representation is selected from a small set invariant states.
2) Associations are updated on a regular timescale, and each associative link acts as an independent Bayesian learning process (i.e. a Bayesian network), giving the links in the semantic network weights relating to

[10]Current programming languages do not make it easy to export the contextual intent behind code or user messages. In future, such an innovation will become essential to the scaled monitoring of a software society.

their importance and frequency of visitation. This applies at each recursive scale of concept aggregation.

The intentional aspects of any system, which was designed or evolved for a niche purpose, are central to the encoding of its semantics. If such information is not captured at the source, it is simply lost. Users of the software may later reinterpret this intent, in the light of current assumptions, but the intended meaning ultimately originates from a source which matches two promises: the 'promise/intent offered' and 'promise/intent accepted' (like lock and key). The receiver always has the last word in interpretation however [8].

C. Software implementation

Prototype code for creating and parsing the knowledge representation, is shared under the Cellibrium [11] project. This employs the simplest possible proof of concept, avoiding programming concerns in favour of pedagogy. Concepts are represented as directory names, each containing subdirectories for the 4+1 association types; then, in turn, a file of association annotations, including timestamps and weights for Bayesian updating. Although filesystem limitations make this approach unsuitable in the long run, it made the coding of a prototype pleasantly free of dependencies and obscure APIs. Everything could be handled with fast, efficient POSIX system calls[11]. All concepts have directories under some root node, which contain subdirectories corresponding to $\pm$ the four spacetime association types. Under these lie files, which document the concepts that may be reached by following that type of association:

```
root/
root/StartConcept/
root/StartConcept/1-4/
root/StartConcept/1-4/NextConcepts*
```

The 'next concepts' are files containing the specific annotations, timestamps, and learning weights of the associations between the starting concept and the chosen next concept.

To frame context, a separate control channel is used, mainly for implementation convenience. The context association is a spacetime containment type (ST3), but its functional role is different to other groupings. Contexts work in exactly the same way as other concepts, with composites and sub-clusters. However, contexts arise *ad hoc* and thus their clusters do not always have lasting meaning as composite concepts. As running catalogues of what is currently going on, somewhat analogous to a browser history, so relevant words pertaining to the agent state were simply strung together in no particular order. This is analogous to the original CFEngine classes [9], where context was a collection of class strings that were probed from the environment by specialized software sensors. Combinations of these classes could be constructed as quasi-logical expressions (acting as hubs) to activate certain concepts (in this case configurations). The basic atoms were drawn from system variables.

In more general cases, at a higher level of abstraction, context must include the 'state of mind' of the cognitive system over the recent past (again, think of browser history), which encapsulates intent as well as environmental impulses. These will be expressible in terms of higher level concepts[12]

IX. Results

A. Assessment

Assessing the success of the prototype 'brainstorming' experiments is not an easy matter, given the qualitative nature of the results. This is not so much a weakness of the experiment, as the nature of the problem. Although it is possible to manufacture artificial metrics by the collection of numerical data, e.g. about numbers of proposed stories, or conceptual overlap with context along a path, etc, these measures are themselves *ad hoc* and largely unhelpful, as they do not correlate easily with the subjective sense of 'a ha!' one has when finding a helpful outcome. So, we are left with a kind of 'Turing test' assessment approach, i.e. 'it looks ok' to a human. Later, once one applies the approach to a specific problem, where target goals can be incorporated from the beginning, we should be able to do better in establishing consistent criteria. I shall not address this further in this report.

B. Data

Data sets used consisted of hundreds to thousands of concept tuples, collected from a variety of sources. These numbers are quite small for a realistic application, but suffice to test some comparative aspects of the approach. Association data were generated by annotating system monitoring in realtime, from software outputs, documentation scans, and by manual knowledge documentation of hints.

In early trials, realtime monitoring data were fed into a knowledge representation in a way that retained a lot of specific information, to see if this might stabilize into a comprehensible emergent pattern. However, this experiment quickly revealed that the potential burden of remembering the detailed semantics of every moment.

If each new measurement set leads to a different context, i.e. making time itself the generator of context, the number of concepts (or amount of data)

[11] It would seem as though a graph database would be the ideal candidate for representing such data, but thus far, the input and query languages for such databases are clumsy and ill-suited to this kind of structure.

[12] Framing and limiting a context is a very hard pedagogical problem. As a semantic spacetime, a knowledge representation is somewhat similar to a higher dimensional field theory with an ultraviolet cutoff, as information is discrete and finite. A story is like a transition function, or S-matrix element, whose vertex rules are based on the causal associations and their propagation rules.

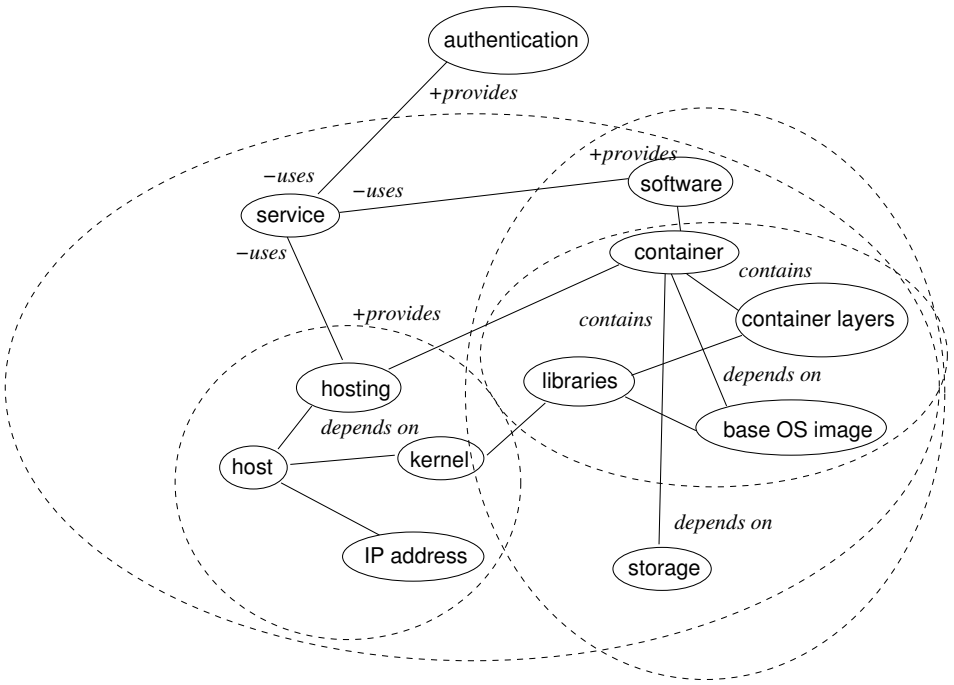

Fig. 17: Software associations used to propose a template for scanning software services. From this model template, a distributed software crawler could be written in a user-secure manner to extract operational knowledge about deployed software. This cluster of concepts illustrates how one can build a template for a core ontology within a small domain.

would be prohibitively large (not for a computer to remember, but for any human or algorithm to make sense of). One cannot escape the fact that our human brains have builtin limitations.

Since we are dealing with semantic interpretations, we cannot simply say 'big data', and treat every knowledge problem as a forensic investigation, because we cannot guarantee that a focused set of concepts will emerge from the data. That is a different problem than having an expert on hand who is already au fait with the necessary knowledge. Thus, we must look for invariant representations of big data, cumulatively over time, that may be interpreted by only a *small* number of (possibly composite) concepts, for the sake of comprehension. Fortunately, studies have shown that meaningful patterns from system monitoring behaviours are few in number, compared to a bulk of non-meaningful noise [30].

C. Structure

Throughout early experiments, the output of a brainstorming episode, based on a given starting concept, tended to lead to either no stories or too many stories. The reasons for this depended on both the structure of data and the selection criteria. Initial attempts to find quality stories were hampered simply by an absence of data. The first trials, using data types to represent contexts, were relatively unsuccessful. The input of knowledge took a huge manual effort to curate, both in terms of syntactic and semantic burdens placed on the author; then, even if someone could be persuaded to undertake encoding, it would be fragile and under-connected. Very little unexpected emergence was possible, as the typing made data overconstrained. The sparseness and fragility of the typed structure meant that concepts could not easily be discovered by free association, and most searches resulted in no results.

Topic map inspired structures [23] created mainly led to trivial stories, so the project was set aside for almost 4 years. Following this hiatus, the models based on topic maps were abandoned [3], and ideas from the promise theoretic approach [5] were revived.

The removal of 'types' as discriminators, and their replacement with context-free lexical terms, led to a large increase in the density of stories. It is difficult to put numerical measures to the improvement, because a radical restructuring of many aspects was involved in this shift, and the results are very dependent on the particular characteristics of example data, which were not preserved over the intervening years between the difference, although an attempt was made to reconstruct similar data. At best one could say that there was a qualitative improvement in the results.

The earliest attempts at discovering emergent connections, were implemented by my collaborator Alva Couch [4], [31], and used a shortest path approach to selecting a unique 'route' between two concepts: initial and final boundary conditions. This is slightly different from full brainstorming, because it is already more constrained by having a definite target concept.

Ultimately, I abandoned the shortest path approach as it creates a tension between quantitative metrics and dynamical selection criteria rather than semantic relationships. The idea that a single path, based on its

length, could be a correct answer to the exclusion of others does not well motivated; although, it is plausible that, given a set of semantically valid paths from A to B, the shortest one might have interesting qualities. The advantages of fixing boundary conditions at the start and the end of a story were compelling, but it is not always so clear what we are looking for when exploring causes and interpretations.

D. Computational complexity

The derivation of the four irreducible spacetime association types brought a significant simplification to the algorithm for exploring locales. Initially a linear search was needed to identify matching concepts from a number of type 'namespaces'. Then associations could be followed, but the question of whether the associations could propagate or not was not encoded directly into the association, requiring another lookup, or so-called inference rule. The possible complexity was of the order of the product of the length of the inference table with the number of outgoing associations (the out-degree of the graph). Thus the pre-classification according to the spacetime 'compass types' led to a reduction in the order of magnitude of the computational complexity by eliminating the need for the inference table. The propagation of these types in inferences is defined by the spacetime classifications themselves [7], and no matching is required.

A further reduction in the degree of 'grammatical complexity' may come from the aggregation of multiple concepts into aggregate classes. If associations can be made to larger umbrella concepts, rather than concepts, which are too specific, then the cost of recursion can be spared. Thus, what may seem to be a context-free expression, linguistically, may be sufficiently represented by a regular language expression. See [5], [7] for a discussion of the grammatical complexity of spacetime associations.

The addition of context, for either positive or negative matching, is the most intensive computation remaining; however, this observation seems critical to establishing semantic relevance of stories. Context, as modelled in the original CFEngine representation [9], [32], and its derived Cellibrium implementation [11], is a non-ordered linear list composed of many atomic characterizations. Each evaluation of context requires parsing the entire list for matches with a similar non-ordered expression composed of the same atoms. The complexity could be of the order of the product of the current context lookup and the learned context lookup, for each association. Supposing that each could be of logarithmic order, by hashing, this is much better than a model without the encoding of spacetime types. The size of a context cannot be pre-determined, because it fluctuates over short times, and grows during long-term learning.

Each link in a story has a complexity cost, associated with the relevance of the link. This was measured by the size of the overlap between current context and learned context (analogous to the mutual information transfer [33], [34]). In the CFEngine model, context was defined as a 'logical boolean expression', which enabled contextual relevance to be reduced even further to a single expression evaluation, whose overlap was generally quite small, i.e. only the length of a very specific boolean expression [32]. There are many ways to optimize such contextual matches, using aggregate classifications and caching, largely because each execution of a CFEngine context is a frozen moment in time. The nature of a cognitive system, on the other hand, is that context is continuously changing, so it will always be more expensive to compute.

E. Sensemaking of stories from structure

Having enough brainstorming paths to analyze, in the output of a search, was an advantage of the new spacetime approach to knowledge representation. Two problems remained however, whose remnants may still be seen in the figures:

- Stories were often unreadable and bizarre, as even valid lexical phrases coughed up along pathways of the graph had little meaning without the original intent of the associations and names preserved and represented. The loss of intent seemed to lie at the heart of this. Intent acts itself as a form of context.
- Often, stories did not terminate, without an artificial length cutoff, in spite of loop detection methods on a per-story basis, as the simplistic recursion allowed stories to contain one another as sub-parts by the free association. This suggested that success may lie in the use of context to constrain stories and bring focus to the results.

Later, when repeated visitation of a concept was forbidden across all stories (as a global constraint), the stories did become finite; however, now an argument could be made that the search was now overconstrained, and ad hoc choices might exclude certain combinatoric stories from being visited at all[13].

In fact, the non-termination of stories is another indication that the graph is well connected, even with the small data sets used, but they were also an artefact of the open-endedness of the brainstorming process. Without a clear goal either in terms of context or specific concept, there is no deterministic criterion for running out of combinatoric options.

The recursive scaling of ideas anchors ideas to context in a more powerful way of attaching intended context than typing. This is basically analogous to a transition to a schemaless database. After the introduction of the recursive hub structure, and the automation API, in figure 14, the story density was

[13]The original approach used by CFEngine was to simulate the 'neuronal dead-time' into concepts, eliminating loops by virtue of concepts having fired already [39]. This worked in CFEngine, because of its simplistic linear processing approach. In a network with multidimensional causation, the approach could easily prevent important answers from emerging rather than protect against repetition.

```
host£ ./stories -s "Cannot mix CIDR notation with xxx-yyy range notation *" -c "software fault" -t 2
0:follows)  "Cannot mix CIDR notation with xxx-yyy range notation *" may be caused by "network policy"
        (in the context of errors and faults)
1:apprxnr)    "network policy" contains "interfaces configuration" (in the context of software)
0:follows)  "Cannot mix CIDR notation with xxx-yyy range notation *" may be caused by "FuzzySetMatch"
         (in the context of errors and faults)
1:preceds)    "FuzzySetMatch" may be used by "iprange" (in the context of software)
2:follows)     "iprange" depends on "network pattern matching" (in the context of software)
0:follows)  "Cannot mix CIDR notation with xxx-yyy range notation *" may be caused by
        "incorrect use of network pattern matching" (in the context of errors and faults)
1:hasprop)    "incorrect use of network pattern matching" is related to "network pattern matching" (in the context of errors and faults)
2:preceds)     "network pattern matching" partly determines "iprange" (in the context of software)
3:follows)      "iprange" may use "FuzzySetMatch" (in the context of software)
3:apprxnr)      "iprange" belongs to or is part of "CGNgine class function" (in the context of software)
4:apprxnr)       "CGNgine class function" belongs to or is part of "system policy" (in the context of software)
4:apprxnr)       "CGNgine class function" belongs to or is part of "CGNgine functions" (in the context of software)
5:apprxnr)        "CGNgine functions" belongs to or is part of "CGNgine policy language" (in the context of software)
3:cntains)      "iprange" has the role of and expresses an attribute "host classifier" (in the context of software software)
4:expr-by)       "host classifier" also known as "class or context label" (in the context of system monitoring)
```

Fig. 18: A brainstorming rooted in an error message generated in a CGNgine system log. This story frames the possible cause(s) of the initial error message anchor point nicely, and pin-points the offending function 'iprange'.

largely unchanged, but the readability of the stories was greatly improved, allowing the kinds of outputs shown in the figures. The structure led naturally to the separation of context into interior and exterior memory channels as mentioned in section VI-F. The interior memory context, in particular, brought immediate comprehension, even when the strict linguistic form was contrived. This, of course, is more a testament to human linguistic acumen than to the skill of a very simple algorithm, but it indicates that the essence of language is captured by a recursive identification of irreducible types.

F. Outputs

Figures 18-22 show typical samples of outputs from a brainstorming. The actual outputs are much longer than these. This print format is not well suited to show the data. The left hand margin shows the spacetime association types (ST1-ST4) followed to make the association, and the indentation shows recursion depth along a single path.

In figure 18, the data are taken from the CGNgine software, using several data transmutors to scan both the source code, and the system policy. The occurrence of an obscure error message in a system output can immediately be tied to the likely sources and points of expression, allowing a user to trace the source all the way back to their intended policy, and through the software layers.

Figure 19 shows data acquired by a software build pipeline. It shows how deployed software depends on its containerized packaging, software dependencies, and host operating system base, all the way down to the location of the deployment. This chain of dependencies may be of use to developers, operations, and risk analysis personnel, to name a few. Further analysis based on the specific semantics could be automated further. Simply gathering this information into a single place, in a readable format is no small task.

Figure 20 shows a more straightforward recursive decomposition of library dependencies discovered by using the Linux `ldd` command. The deeply nested and extensive list of cross dependencies is quite impossible for any human to comprehend, but is straightforward to encode for analysis.

Figure 21 shows a facsimile of the distributed system used by patients to register with a new doctor in Norway, as curated from the perspective of a domain expert. The procedure involves going to a website, but first one must have a number of credentials from third party sites, totally unrelated to the public service. This kind of constellation of independent services, integrated into a meaningful whole would be handled by some kind of wizard software if it were a task for a single computer. In the web era, there is no single agency with the responsibility to integrate this information. A cognitive system, which monitors services in a smart city, could do this, for instance.

Figure 22 shows some more playful examples of domain knowledge, concerning the purpose of different services, in different contexts. A lexical token like 'tidal' might refer to many different things in different contexts. Here the brainstorming is able to distinguish these cases using the recursive structure of the matroid hubs. In a more sophisticated rendition of the brain dump, one would be able to select specific interpretations from these structures without risk of conflict. Whether the source of the semantics is manual curation or trained natural language processing, it does alter the key benefit of the approach used here: namely that it is orders of magnitude cheaper to build and to use than an approach based on repeated use of 'big data'. Such approaches have been used to discover vector congruences and emergent functional semantics, but only with extensive training [35]–[38]. Such approaches could still be used as inputs to the present approach, in the absence of something more practical.

```
host$ ./stories -s "microservice"
```

0:apprxnr) "microservice" is approximately "service" (in the context of software ops operations develop write)
1:follows) "service" may use "user authentication" (in the context of software ops operations develop write)
2:hasprop) "user authentication" has the role of "authentication" (in the context of software ops operations develop write)
and also note "authentication" is a role fulfilled by "service authentication" (in the context of software ops operations develop write)
1:follows) "service" may use "service authentication" (in the context of software ops operations develop write)
2:hasprop) "service authentication" has the role of "authentication" (in the context of software ops operations develop write)
and also note "authentication" is a role fulfilled by "user authentication" (in the context of software ops operations develop write)
1:follows) "service" depends on "software SOFTWARENAME" (in the context of software ops operations develop write)
2:follows) "software SOFTWARENAME" depends on "container CONTAINERNAME" (in the context of software ops operations develop write)
3:follows) "container CONTAINERNAME" depends on "package PACKAGENAME2 VERSION2" (in the context of software ops operations develop write)
4:hasprop) "package PACKAGENAME2 VERSION2" has the role of "package" (in the context of software ops operations develop write)
and also note "package" is a role fulfilled by "package PACKAGENAME1 VERSION1" (in the context of software ops operations develop write)
and also note "package" is a role fulfilled by "package PACKAGENAME3 VERSION3" (in the context of software ops operations develop write)
4:follows) "package PACKAGENAME2 VERSION2" depends on "library LIB1" (in the context of software ops operations develop write)
5:preceds) "library LIB1" partly determines "package PACKAGENAME1 VERSION1" (in the context of software ops operations develop write)
6:hasprop) "package PACKAGENAME1 VERSION1" has the role of "package" (in the context of software ops operations develop write)
and also note "package" is a role fulfilled by "package PACKAGENAME3 VERSION3" (in the context of software ops operations develop write)
5:preceds) "library LIB1" partly determines "package PACKAGENAME3 VERSION3" (in the context of software ops operations develop write)
6:hasprop) "package PACKAGENAME3 VERSION3" has the role of "package" (in the context of software ops operations develop write)
and also note "package" is a role fulfilled by "package PACKAGENAME1 VERSION1" (in the context of software ops operations develop write)
1:follows) "service" depends on "hosting" (in the context of software ops operations)
2:expr-by) "hosting" is a role fulfilled by "hosting CONTAINERNAME INSTANCE" (in the context of software ops operations develop write)
2:follows) "hosting" depends on "host HOSTNAME" (in the context of execute run software ops operations)

Fig. 19: Knowledge scanned from a distributed application (simulated) shows the structure of the service, its software and dependencies, from containers to operating system package names. The litany of components and service dependencies might be used for troubleshooting or risk/vulnerability/impact analysis.

G. Inadequate context

During the development of this approach, the notion of context was subject to a lot of confusion and misdirection. The way logic-based approaches to semantic modelling have tried to use context (as data types) turned out to be quite wrong: there exists no objective version of knowledge that can determine a fixed type-hierarchy or preferred ontology. Knowledge is very much in the eye of the beholder, and every observer forms an individual ontology [3]. Context is not an objective categorization, it is a running state of mind (like a browser search history that records our intent). As long as we try to treat a knowledge representation as an objective static boundary condition on knowledge, rather than as a private subjective interpretation, we will run into difficulties.

In this present approach, motivated by spacetime classification of semantics, and which thus far ignores any preferred ordering by link weights, semantic stability is encouraged simply by a structural aggregation of concepts, analogous to human language. So, even if there are random variations, using promises like 'A is near B' effectively allows us to collapse disconnected or multiple stories into a single broader line story. What is important to reasoning is that there lies a causal set of paths, that may be integrated with the specific boundary conditions from start to end[14].

X. Summary and remarks

The recent focus of work in artificial intelligence (AI) has been on the training of artificial neural networks to recognize increasingly complicated patterns, like voice, faces and handwriting. Trained networks mimic human cognitive sensory systems, usually one by one. Neural networks are trained by attempting to replay a lifetime of experiences at high-speed into a learning process. The extent to which such a network accumulated, and made predictive, depends on the extent to which its spatial structure can mimic a representation of the semantics one happens to be looking for, with only a single trigger [7]. However, it is impossible to measure the certainty of even this simple kind of reasoning, indicating that an entirely separate stage of processing by a more deterministic representation is needed for reasoning that goes beyond a single inference. That is where the present work fits in. The recognition of the lexicon of basic patterns is the first part of sensory perception, the second part is associative reasoning based on its tokens.

[14]This is tantalizingly reminiscent of a quantum path integral, for reasons the reader may ponder without suggestion from me.

```
0:follows)  "application cgn-agent" depends on "cgn-agent" (in the context of host configuration maintenance agent CGNgine)
1:follows)    "cgn-agent" depends on "libz.so.1" (in the context of host application software dependencies security)
2:follows)      "libz.so.1" depends on "libc.so.6" (in the context of host application software dependencies security)
3:follows)        "libc.so.6" depends on "!lib64!ld-linux-x86-64.so.2" (in the context of host application software dependencies security)
4:preceds)          "!lib64!ld-linux-x86-64.so.2" partly determines "libkrb5support.so.0" (in the context of host application software de-
pendencies security)
5:follows)        "libkrb5support.so.0" depends on "libpcre.so.1" (in the context of host application software dependencies security)
6:follows)          "libpcre.so.1" depends on "libpthread.so.0" (in the context of host application software dependencies security)
7:preceds)        "libpthread.so.0" partly determines "libdbus-1.so.3" (in the context of host application software dependencies security)
8:preceds)          "libdbus-1.so.3" partly determines "libavahi-client.so.3" (in the context of host application software dependen-
cies security)
9:follows)          "libavahi-client.so.3" depends on "libdl.so.2" (in the context of host application software dependencies security)
10:preceds)       "libdl.so.2" partly determines "libcrypto.so.1.0.0" (in the context of host application software dependencies security)
10:preceds)         "libdl.so.2" partly determines "libpq.so.5" (in the context of host application software dependencies security)
    ++ more ....
10:preceds)       "libdl.so.2" partly determines "libp11-kit.so.0" (in the context of host application software dependencies security)
10:preceds)         "libdl.so.2" partly determines "libgssapi˙krb5.so.2" (in the context of host application software dependen-
cies security)
    ++ more ....
10:preceds)       "libdl.so.2" partly determines "libgcrypt.so.20" (in the context of host application software dependencies security)
    ++ more ....
9:follows)          "libavahi-client.so.3" depends on "libavahi-common.so.3" (in the context of host application software depen-
dencies security)
10:preceds)           "libavahi-common.so.3" partly determines "libvirt.so.0" (in the context of host application software depen-
dencies security)
    ++ more ....
9:preceds)          "libavahi-client.so.3" partly determines "libvirt.so.0" (in the context of host application software dependen-
cies security)
10:follows)       "libvirt.so.0" depends on "libcrypto.so.1.0.0" (in the context of host application software dependencies security)
    ++ more ....
10:follows)         "libvirt.so.0" depends on "libidn.so.11" (in the context of host application software dependencies security)
10:follows)         "libvirt.so.0" depends on "libudev.so.1" (in the context of host application software dependencies security)
10:follows)         "libvirt.so.0" depends on "libnl-3.so.200" (in the context of host application software dependencies security)
10:follows)         "libvirt.so.0" depends on "libp11-kit.so.0" (in the context of host application software dependencies security)
    ++ more ....
8:preceds)        "libdbus-1.so.3" partly determines "libvirt.so.0" (in the context of host application software dependencies security)
9:follows)          "libvirt.so.0" depends on "libcrypto.so.1.0.0" (in the context of host application software dependencies security)
10:follows)         "libcrypto.so.1.0.0" depends on "libdl.so.2" (in the context of host application software dependencies security)
    ++ more ....
10:preceds)           "libcrypto.so.1.0.0" partly determines "libpq.so.5" (in the context of host application software dependen-
cies security)
    ++ more ....
```

Fig. 20: Dependency relationships obtained by a simple scan of a process container are a hopeless task for a human to trace. The level of interdependency is very large. The ability to trace dependencies across software deployed in multiple distributed locations could be central to addressing transmission of faults and attacks. We see that context strings can become quite long. This is both expected and welcome, as it increases the possibility to filter and conjoin relationships based on specific circumstances, moving away the the kind of general brain-dumps shown here.

In this report, the idea is not to mimic human faculties but to integrate them, across multiple scales of experience, allowing us to to add selected semantic, data of high quality, incrementally from any source to a single model with very few limitations. The learning described here is unsupervised, in the AI sense, but still curated by a surrounding 'society' of expert interactions, so it is supervised by an emergent framework of domain expertise. No learning can happen without some external selection process, just as no child can be programmed with life skills without others to raise it. The boundary between raw exterior and curated interior knowledge is what cognition enables: to be able to represent and pass on knowledge is a compressed (tokenized) form for consumption by others, without the need to undergo every experience, blow by blow, and in person.

These experiments are very interesting, but there are plenty of issues to address in future work. Finding how to constrain stories and bring focus to their trains of thought is the major problem. With the use of a spacetime inspired structure, stories are no longer as scarce as when trying to use an overconstrained typed approach. Lateral reasoning is now possible, and relevance is maintained by representing context. The treatment of context in this note is weak, but there is much room for improvement. In [10], I speculated on the role of emotional state in determining context. Emotional weight plays a significant role in cognition for setting context and interpretation priorities. From the viewpoint in [7], emotions seem to play the role of very coarse aggregate 'concepts' stimulated by sensory/introspective inputs. Emotional context is thus a systemic assessment of some 'agent' based on a number of contextual factors. It is a threshold judgement, based on policy:

```
host$ ./stories -t 2 -s "doctor service" |more
Found story subject: "doctor service"
Stories of type 2 only

0:follows)  "doctor service" depends on "patient appointment" (in the context of need to see a doctor)
1:follows)    "patient appointment" depends on "patient doctor register binding" (in the context of need to see a doctor)
2:follows)      "patient doctor register binding" depends on "doctor patient binding service" (in the context of patient doctor registration)
3:follows)        "doctor patient binding service" depends on "patient authenticated" (in the context of patient doctor registration)
4:follows)          "patient authenticated" depends on "identity credentials" (in the context of identity authentication verification)
5:follows)        "identity credentials" may originate from "https:!!url1!form!element2" (in the context of identity authentication verification)
5:follows)            "identity credentials" uses step 1 and may originate from "https:!!url1!form!element1" (in the context of identity authentication verification i
dentity authentication verification)
5:follows)            "identity credentials" uses step 1 "https:!!url2!form!element2" (in the context of identity authentication verification)
5:preceds)            "identity credentials" are required for and partly determines "doctor authenticated" (in the context of identity authentication verification ide
ntity authentication verification)
6:preceds)             "doctor authenticated" is required for and partly determines "accepted doctor patient binding" (in the context of patient doctor registration
patient doctor registration)
7:follows)               "accepted doctor patient binding" depends on "doctor authorized" (in the context of patient doctor registration)
7:follows)               "accepted doctor patient binding" depends on "doctor" (in the context of patient doctor registration)
7:follows)               "accepted doctor patient binding" depends on "patient" (in the context of patient doctor registration)
4:preceds)          "patient authenticated" partly determines "have public health service access" (in the context of need to visit a doctor)
5:follows)          "have public health service access" depends on "public health service available" (in the context of need to visit a doctor)
6:follows)                "public health service available" depends on "general practictioner doctor available" (in the context of need to visit a doctor)
6:follows)            "public health service available" depends on "open for business" (in the context of need to visit a doctor)
6:preceds)            "public health service available" partly determines "public health service" (in the context of need to visit a doctor)
7:follows)              "public health service" depends on "patient uses appointment" (in the context of need to visit a doctor)
7:follows)              "public health service" depends on "doctor availability" (in the context of need to visit a doctor)
3:follows)        "doctor patient binding service" depends on "doctor authenticated" (in the context of patient doctor registration)
4:follows)          "doctor authenticated" depends on "identity credentials" (in the context of identity authentication verification)
5:follows)        "identity credentials" may originate from "https:!!url1!form!element2" (in the context of identity authentication verification)
5:follows)            "identity credentials" uses step 1 and may originate from "https:!!url1!form!element1" (in the context of identity authentication verification i
dentity authentication verification)

Total independent outcomes/paths = 28
Estimated relevance of outcomes 0/0
```

Fig. 21: Cooperating distributed (micro)services may have no cohesive storyline to connect them, without aggregating the intended relationships between them. Intentional (promise oriented) documentation allows us to generate a kind of documentation 'wizard' in realtime from processes changing in realtime. Causal links show the representation of a complicated distributed system, which could be the basis of a wizard for integrating apparently unrelated parts in a single view.

	POSITIVE	NEGATIVE
INTERIOR	CONTENTED	DISTRESSED
EXTERIOR	LIKE	DISLIKE
PREDICTION	OPTIMISTIC	PESSIMISTIC

States like distress are easy to measure in a finite system. When a threshold of behaviour is reached and a system is unable to keep its promises, e.g. the thrashing to empty a queue. Emotions could be semanticized versions of these major conditions.

States like good, bad, danger, happy, sad, etc are components of what we would think of as emotional states. These give clear meanings to other measurements: how we respond to a context (state) and the associations we make during good and bad times affect how we recall concepts later. If we feel strongly about something (good or bad) this translates into an importance rank of an association. We want to sum the recurrence scores of the concepts to label their importance. It is possible that we might even be able to make all of these by combining the simultaneous activation of senses with the concepts of good and bad: e.g.

$$\text{Good/bad} \cap \text{person} \rightarrow \text{love/hate} \quad (13)$$

$$\text{Good/bad} \cap \text{senses} \rightarrow \text{happy/sad} \quad (14)$$

$$\text{Good/bad} \cap \text{comparison} \rightarrow \text{true/false} \quad (15)$$

This remains a topic for future exploration. I hope to return to all these issues, in the context of applications, in future work.

Acknowledgement: I am deeply indebted to Steve Pepper for in depth discussions about linguistics and knowledge representations. I am also grateful to Nikos Anerousis, Anup Kalia, Maja Vukovic, and Jin Xiao for helpful conversations.

REFERENCES

[1] M. Burgess. *In Search of Certainty: the science of our information infrastructure*. Xtaxis Press, 2013.

[2] Mark Burgess. Knowledge management and promises. *Lecture Notes on Computer Science*, 5637:95–107, 2009.

```
host$ ./stories -s "microservice"
```

0:hasprop) "microservice" has the role of "application" (in the context of software ops operations develop write)
and also note "application" is a role fulfilled by "application cgn-agent" (in the context of host application software dependencies security)
and also note "application" is a role fulfilled by "application service" (in the context of software ops operations develop write)
1:apprxnr) "application" has instance "tidal" (in the context of online services)
2:follows) "tidal" depends on and NOT depends on "facebook" (in the context of online services applications physics gravity orbit moon satellite distortion)
3:expr-by) "facebook" offers service "facebook authentication" (in the context of logging in)
3:cntaind) "facebook" offers service "facebook authentication" (in the context of logging in)
2:follows) "tidal" is authenticated by "facebook authentication" (in the context of online services applications)
3:hasprop) "facebook authentication" is a service offered by "facebook" (in the context of logging in)
3:cntains) "facebook authentication" is a service offered by "facebook" (in the context of logging in)
and also note "tidal" is generalized by "gravitational effect" (in the context of physics gravity orbit moon satellite distortion)
2:cntains) "tidal" is generalized by "gravitational effect" (in the context of physics gravity orbit moon satellite distortion)
3:follows) "gravitational effect" depends on "distance" (in the context of physics)
3:follows) "gravitational effect" depends on "mass" (in the context of physics)
1:cntaind) "application" has instance "tidal" (in the context of)
2:follows) "tidal" depends on and NOT depends on "facebook" (in the context of online services applications physics gravity orbit moon satellite distortion)
3:expr-by) "facebook" offers service "facebook authentication" (in the context of logging in)
3:cntaind) "facebook" offers service "facebook authentication" (in the context of logging in)
2:follows) "tidal" is authenticated by "facebook authentication" (in the context of online services applications)
3:hasprop) "facebook authentication" is a service offered by "facebook" (in the context of logging in)
3:cntains) "facebook authentication" is a service offered by "facebook" (in the context of logging in)
2:apprxnr) "tidal" is generalized by "gravitational effect" (in the context of physics gravity orbit moon satellite distortion)
3:follows) "gravitational effect" depends on "distance" (in the context of physics)
3:follows) "gravitational effect" depends on "mass" (in the context of physics)
and also note "tidal" is generalized by "gravitational effect" (in the context of physics gravity orbit moon satellite distortion)

Fig. 22: Knowledge scanned from a distributed application (simulated).

[3] M. Burgess. *New Research on Knowledge Management Models and Methods.*, chapter What's wrong with knowledge management? The emergence of ontology. InTech, 2012.

[4] A. Couch and M. Burgess. Compass and direction in topic maps. *(Oslo University College preprint)*, 2009.

[5] M. Burgess. Spacetimes with semantics (i). `http://arxiv.org/abs/1411.5563`, 2014.

[6] M. Burgess. Spacetimes with semantics (ii). `http://arxiv.org/abs/1505.01716`, 2015.

[7] M. Burgess. Spacetimes with semantics (iii). `http://arxiv.org/abs/1608.02193`, 2016.

[8] J.A. Bergstra and M. Burgess. *Promise Theory: Principles and Applications*. $\chi tAxis$ Press, 2014.

[9] M. Burgess. A site configuration engine. *Computing systems (MIT Press: Cambridge MA)*, 8:309, 1995.

[10] M. Burgess. Computer immunology. *Proceedings of the Twelth Systems Administration Conference (LISA XII) (USENIX Association: Berkeley, CA)*, page 283, 1998.

[11] M. Burgess. Cellibrium project. `https://github.com/markburgess/Cellibrium`, 2015.

[12] Stephen Tratz and Eduard Hovy. A taxonomy, dataset, and classifier for automatic noun compound interpretation. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 678–687. Association for Computational Linguistics, 2010.

[13] M. Curd and J.A. Cover, editors. *Philosophy of Science: The Central Issues*. Norton, 1998.

[14] J. Topping. *Errors of Observation and their Treatment*. Chapman and Hall, 1972.

[15] R.H. Dieck. *Measurement and Uncertainty (Methods and Applications)*. Instrument, Systems and Automation Society, Triangle Park, North Carolina, third edition edition, 2002.

[16] J. Schwinger. *Quantum Kinematics and Dynamics*. Addison Wesley, 1970.

[17] D.B. West. *Introduction to Graph Theory (2nd Edition)*. (Prenctice Hall, Upper Saddle River), 2001.

[18] C. Berge. *The Theory of Graphs*. (Dover, New York), 2001.

[19] D.J. Watts. *Small Worlds*. (Princeton University Press, Princeton), 1999.

[20] M.B. Moser, D.C. Rowland, and E.I. Moser. Place cells, grid cells, and memory. *Perspectives in Biology*, 2016.

[21] G. Deutsche. *The Unfolding of Language*. Academic Press, 2005.

[22] R.W. Langacker. *Cognitive Grammar, A Basic Introduction*. Oxford, Oxford, 2008.

[23] S. Pepper. *Encyclopedia of Library and Information Sciences 4th Ed*, chapter Topic Maps. CRC Press, ISBN 9780849397127, 2015.

[24] M. Burgess, G. Canright, and K. Engø. A graph theoretical model of computer security: from file access to social engineering. *International Journal of Information Security*, 3:70–85, 2004.

[25] Qifan Wang, Dan Zhang, and Luo Si. Semantic hashing using tags and topic modeling. In *Proceedings of the 36th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '13, pages 213–222, New York, NY, USA, 2013. ACM.

[26] M. Tomasello. *Constructing a Language: A Usage-Based Theory of Language Acquisition.* Harvard University Press, 2003.

[27] Z. Chen, N. Ma, and B. Liu. Lifelong learning for sentiment classification. In *Proceedings of the 53 Annual meeting of the Association of Computational Linguistics*, pages 26–31, 2015.

[28] Z. Chen and B. Liu. Topic modeling using topics from many domains, lifelong learning and big data. In *Proceedings of the 31st International Conference on Machine Learning*, 2014.

[29] M. Burgess. Probabilistic anomaly detection in distributed computer networks. *Science of Computer Programming*, 60(1):1–26, 2006.

[30] M. Burgess, H. Haugerud, T. Reitan, and S. Straumsnes.

Measuring host normality. *ACM Transactions on Computing Systems*, 20:125–160, 2001.

[31] A. Couch and M. Burgess. Human-understandable inference of causal relationships. In *Proc. 1st International Workshop on Knowledge Management for Future Services and Networks*, Osaka, Japan, 2010.

[32] M. Burgess. A tiny overview of cfengine: convergent maintenance agent. In *Proceedings of the 1st International Workshop on Multi-Agent and Robotic Systems, MARS/ICINCO*, 2005.

[33] C.E. Shannon and W. Weaver. *The mathematical theory of communication*. University of Illinois Press, Urbana, 1949.

[34] T.M. Cover and J.A. Thomas. *Elements of Information Theory*. (J.Wiley & Sons., New York), 1991.

[35] O. Levy and Y. Goldberg. Linguistic regularities in sparse and explicit word representations. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, page 171180, 2014.

[36] T. Mikolov, W. Yih, and G Zweig. Linguistic regularities in continuous space word representations. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT-2013)*, pages 746–751, 2013.

[37] Marek Rei and Ted Briscoe. Looking for hyponyms in vector space. In *CoNLL*, 2014.

[38] Sridhar Mahadevan and Sarath Chandar. Reasoning about linguistic regularities in word embeddings using matrix manifolds. *CoRR*, abs/1507.07636, 2015.

[39] M. Burgess and D. Skipitaris. Adaptive locks for frequently scheduled tasks with unpredictable runtimes. *Proceedings of the Eleventh Systems Administration Conference (LISA XI) (USENIX Association: Berkeley, CA)*, page 113, 1997.