Structural Representation and Reasoning in a Hybrid Cognitive Architecture

John Licato • Ron Sun • Selmer Bringsjord
{licatj, rsun, selmer}@rpi.edu
Rensselaer Polytechnic Institute

***Abstract*—Psychologically and neurobiologically plausible models of knowledge often must make a difficult choice between distributed and localist representation. Distributed representation can come with all of the advantages of neural networks, but localist models allow for structured knowledge to be represented unambiguously and reasoned over in rigorous, transparent fashion. We present a way of representing knowledge within the hybrid cognitive architecture CLARION. Our system allows both structured knowledge and distributed knowledge to synergistically coexist while remaining within the limits defined by CLARION's dual-process framework. After showing how our system can allow more complex knowledge structures to arise, we describe algorithms that use such structures to model many types of reasoning, including: analogical reasoning, deductive reasoning, moral reasoning, and more. We place the structural knowledge afforded CLARION within a formal hierarchy of expressivity for such knowledge, and discuss implications of this work.**

I. Introduction - Distributed, Localist, and Structured Knowledge

In the history of cognitive modeling, perhaps no issue is as centrally important as that of representation. Which form is best to represent the knowledge of the cognitive system? The answers to this by now are well known, and they tend to lean towards one of two camps: distributed or localist. Throughout the literature, similar debates have arisen under many different names—system I vs. system II [12], connectionist vs. logicist [1], and implicit vs. explicit [15] are just a few of the pairs that parallel the distributed vs. localist dichotomy we discuss here;[1] but the general positions which separate each pair remain relatively consistent. Localist systems are mostly symbolic, meaning that one semantic concept corresponds to one atomic unit. Here, a unit typically refers to an indivisible element of the system (for example, a single node in a network), and a concept refers to some distinct semantic element which can often be named by a single word or short phrase. In contrast, distributed systems are subsymbolic, meaning that a single semantic concept might be represented by multiple units. Depending on the implementation, each unit might also correspond to a microfeature, which represents a low-level feature that may not necessarily correspond to an explicit concept, and can typically be extracted through learning algorithms that automatically develop fine-grained internal representations [13]. Such units may also correspond to multiple concepts simultaneously, which may allow for overlap between concepts to be easily detected.

This ability to easily detect overlap is an advantage distributed systems have over strictly localist systems, and there are many others as well. Distributed systems provide knowledge that is associative, content-addressable, and easily parallelizable [5]. Similarity measurements between multiple concepts and processes like similarity-based reasoning are also easier in distributed systems [16]. Localist systems, in turn, provide clarity—knowledge can be represented unambiguously, and relations between elements in localist systems often lend themselves to more understandable semantic interpretations than the links between distributed units.

However, the full power of localist systems does not lie solely in its ability to represent concepts with single units. Rather, the ability for these localist units to be organized in *well-defined structures*, and the ability of a cognitive system to perform operations on and between such structures, are what form the foundation of many abilities regarded to be indispensable to any high-level cognizer. Such abilities include deductive reasoning and analogical reasoning, the two components of Analogico-Deductive Reasoning (ADR), which is utilized by a variety of reasoners from young children [9] to advanced practitioners of mathematical logic [10].

Because the strengths of distributed and localist systems are so disjoint, any cognitive architecture which remains exclusively distributed or localist excludes the ability to effectively model a wide range of human abilities. In light of this fact, there have been some attempts to develop *dual-process* architectures which represent knowledge using explicit (localist) and implicit (distributed) dimensions simultaneously. Such architectures must necessarily focus on the synergistic interaction between the two types of knowledge. This is one of the primary principles behind CLARION [15], the dual-process cognitive architecture which we work with in this paper.

CLARION provides us with a means to represent localist and distributed knowledge, but so far has not demonstrated structured knowledge as would be required by deductive or analogical reasoning. In this paper, we show how such structured knowledge can be represented in CLARION using only mechanisms that have been defined and used in previous papers. We believe the resulting representation scheme is flexible enough to work synergistically with the distributed component of CLARION's knowledge, but at the same

This work was supported by the Office of Naval Research.

[1] This is a loose alignment, since for example not all will agree that localist and logicist systems are the exact same thing.

time is rigorous enough to be used in deductive reasoning and interpreted in a natural manner (e.g. we can convert these structures into readable S-expressions). We will start by summarizing the CLARION system, and then we will describe our additions.

II. CLARION

CLARION [15] is an integrative cognitive architecture that has a dual-process structure consisting of two levels: explicit (top level) and implicit (bottom level). These levels roughly correspond to the localist-distributed split we described earlier. CLARION has been able to model a wide variety of cognitive phenomena while maintaining psychologically plausible data structures and algorithms; this makes it an ideal choice for our purposes. By showing that structured reasoning can emerge from no more than the mechanisms in CLARION which previous literature have already shown to be psychologically plausible, we intend to provide a strong foundation for showing that these new structures are psychologically plausible as well.

The architecture is further divided into four *subsystems*, each with explicit and implicit levels, which specialize in different aspects of cognition: The Motivational Subsystem (MS), the Metacognitive Subsystem (MCS), the Action-Centered Subsystem (ACS), and the Non-Action-Centered Subsystem (NACS). We will be focusing on the NACS in this paper.

A. NACS — the Non-Action-Centered Subsystem

The NACS contains general knowledge about the world that is not contained in the ACS. Whereas the ACS is meant to capture the knowledge that directly causes decision making while interacting with the world, the knowledge in the NACS is often more deliberative and used for making inferences. The top level of the NACS is called the General Knowledge Store (GKS), and it contains localist chunks which can be linked to each other using Associative Rules (ARs).

The bottom level of the NACS is called the AMN, or the Associative Memory Network, and it contains implicit associative knowledge encoded as dimension-value pairs (DV pairs). Each GKS chunk is connected to a set of DV pairs in the AMN with some weight that can be adjusted over time. This unique structure gives CLARION the ability to define a *directed* similarity measure between two chunks c_1 and c_2 which is derived from the amount of overlap between the DV pairs connected to the two chunks [14], [18], [16]:

$$S_{c_1 \to c_2} = \frac{\sum_{i \in c_2 \cap c_1} W_i^{c_2} \times A_i}{f(\sum_{i \in c_2} W_i^{c_2} \times A_i)} \qquad (1)$$

Where $f(x) = x^{1.0001}$. Sun and Zhang (2004) define A_i as the strength of the value of dimension i in chunk c_1, and $W_i^{c_2}$ as the weights of the DV pairs specified with respect to c_2. However, in this paper we will be simplifying things by setting all A and W values to 1, which reduces Equation 1 to a function of the number of dv pairs connected to c_1 and c_2:

$$S_{c_1 \to c_2} = \frac{|c_1 \cap c_2|}{|c_2|^{1.0001}} \qquad (2)$$

Note that it is possible for the denominator in Equation 2 to be zero, in which case the entire equation is given the default value of 1.

The Associative Rules (ARs) link groups of chunks to other chunks in the GKS, and consist of a set of condition chunks $c_1, c_2, ...$ and a single conclusion chunk d. For any given AR, each condition chunk i has a weight W_i such that $\sum_i W_i = 1$. We will write out a single associative rule in the following format:

$$(c_1, c_2, ..., c_n) \Rightarrow d$$

The chunks in the GKS and DV pairs in the AMN have activation levels which can be set by CLARION's other subsystems. Activations can also spread through the NACS using the chunk-DV pair connections and the top-level ARs. The manner in which this activation spreads can be restricted, too: other subsystems can temporarily disable Rule-Based Reasoning (activation spreading through ARs) or Similarity-Based Reasoning (activation spreading through chunk similarity), or perform activation propagation as some weighted combination of both of these reasoning types. These abilities are detailed further in Sun & Zhang (2004, 2006), in which these mechanisms are shown to be psychologically plausible by using them to closely emulate the results of psychological studies. We use no more than these mechanisms to construct the knowledge structures in this paper.

III. Representing Structured Knowledge

The associative rules linking NACS chunks already seem to impart a sort of weak structure to the GKS, but the sort of structure we are looking for has at least four desirable qualities, which are reflective of some of the difficulties in introducing structural knowledge. First, **D1:** the structure needs to be decomposable. It needs to be able to recognize that a certain group of chunks is part of a structure, and given that structure be able to retrieve chunks based on their functional role within that structure.

Closely related is the second quality, **D2:** the structure should be expressible. We want the ability to easily convert well-formed structures into some easy-to-parse format (in this case, S-expressions such as those used by LISP) and vice versa.

Next, **D3:** a structure should not introduce ambiguity. If chunks are a part of multiple, independent structures, the system should still be able to parse the individual structures and treat the chunk just as if they were alone. The *roles* played by a chunk in one part of a structure should not confuse the roles it plays in any other structure (or even in another part of the same structure).

And finally, **D4:** we want clarity. If reasoning processes occur that use these structures, it would be nice to know which structures were used and in what way. This is extremely important for a deductive reasoning system, where

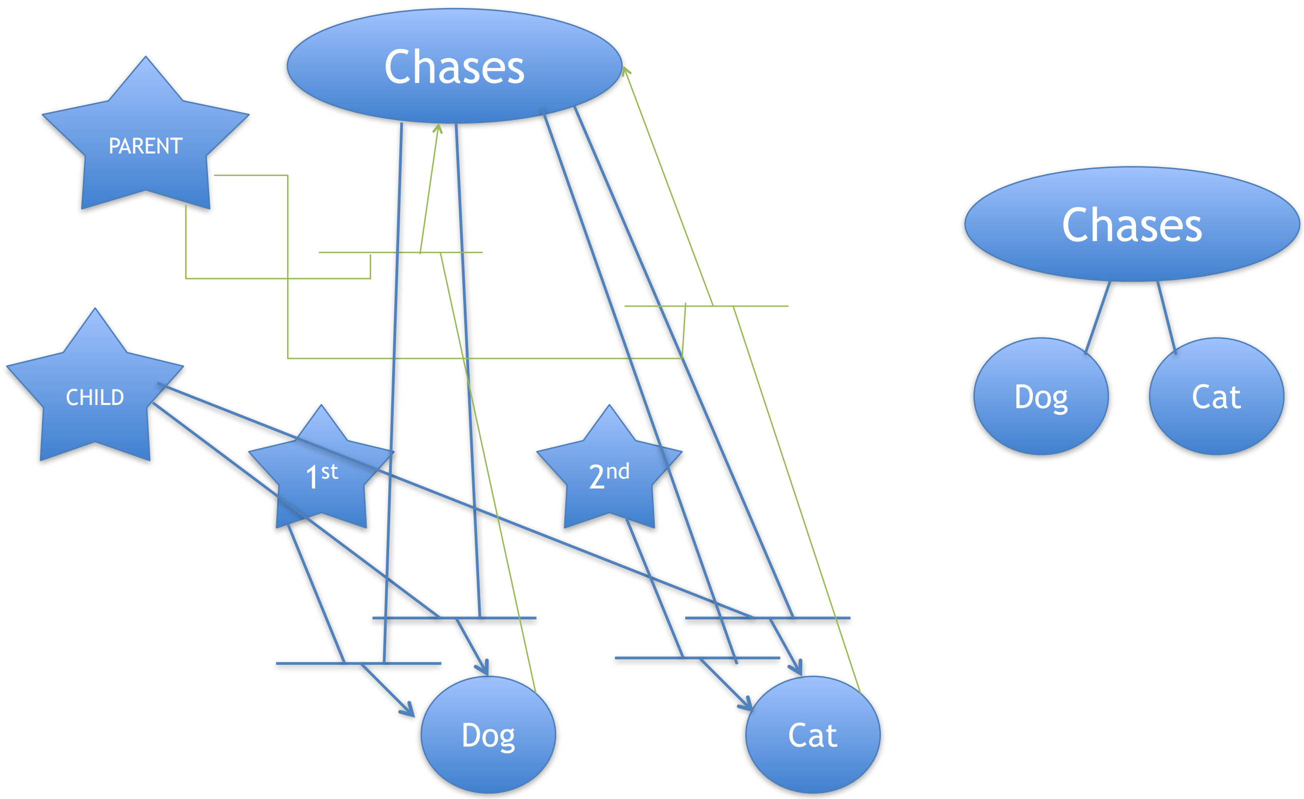

Fig. 1. A knowledge structure representing the proposition $CHASES(DOG, CAT)$. On the right is the simplified version, which omits the CDCs and many of the ARs, though they are there (just not pictured).

every inference step requires proper justification that can be independently verified.

A. Introducing Chunk Types

We will start by assigning *types* to chunks. The scope of these types, however, holds only within the context of a complete structure, for reasons we will explain shortly. Right away we can target desideratum **D2** by basing our structures on the well-established S-expressions, which are perhaps most notable for their use in the logic programming language LISP. An S-expression is of the form:

$$(P \;\; o_1 \;\; o_2 \;\; \dots \;\; o_n)$$

Where P is a predicate, and each o_i is either an object or another S-expression. Our first chunk type, then, is the object chunk. We also define a proposition chunk, which is both a marker of the relationship between the object chunks, and a placeholder for the proposition's predicate symbol. The proposition and object chunks are pictured in Figure 1 as oval shaped objects.

Of course, something needs to link these chunks together, and that is where Cognitively Distinguished Chunks (CDCs) come in. Given that all neurobiologically normal adult humans perform structured reasoning, we should assume that there are some common cognitive abilities which are either innate or develop very early in life which allow for structured knowledge to emerge. CDCs are meant to reflect these abilities, and we maintain psychological plausibility by placing the following restrictions on them. Firstly, CDCs are fixed—We do not define any algorithms that create or destroy CDCs. Secondly, CDCs are known to basic reasoning algorithms. The algorithm we describe later in this paper which performs analogical reasoning, for example, can refer to certain CDCs directly, under the assumption that these are basic features of structured knowledge. Finally, if there is a function that can be easily performed using a CDC, then that function is assumed to be a basic ability of any neurobiologically normal adult human reasoner.

CDCs are depicted as star-shaped (Figure 1). Associative rules link the CDCs to the chunks in the structure. For example, the *PARENT* CDC links object nodes to proposition nodes. In Figure 1, which depicts the proposition *CHASES(DOG CAT)*, the *PARENT* CDC is part of two ARs (depicted in the Figure as an arrow with multiple tails and one head):

$$(DOG, PARENT) \Rightarrow CHASES$$
$$(CAT, PARENT) \Rightarrow CHASES$$

Each object node has a weight of 0.5 within the AR. In fact, for all ARs we mention in this paper, the weight is distributed evenly amongst all objects in the AR's condition unless otherwise mentioned. The "basic ability" that these two ARs correspond to is the ability to recall propositions involving an object, given nothing but that object. Imagine being asked to recall some fact about dogs. Among others, one of the facts that likely would be recalled is that dogs chase cats (assuming, of course, that the reasoner in question is aware of this fact). That is modeled here by activating the *PARENT* and *DOG* chunks. The activation would spread through any ARs which contain those two chunks in their conditions, and the resulting proposition nodes would be activated.

A *CHILD* CDC is also defined to introduce some redundancy into the structure. However, because a single CDC would not satisfy desideratum **D3**, we introduce *Ordinal CDCs*, which are also pictured in Figure 1 as *1ST, 2ND,* etc. Ordinal CDCs simply preserve the roles objects play within propositions in a general way that does not name the roles specifically (contrast this with the LISA model [6], which is an inspiration for much of our work described here. LISA has distinct role units for every type of role.).

The basic proposition structure we have been describing can also be nested, so that instead of an object chunk a proposition chunk can have another proposition chunk as a child. A proposition chunk can even have a single object chunk as a child multiple times, as would be necessary in the proposition $P(a, X, a)$ (Figure 2). We now move on to describing how reasoning can be performed over these structures.

IV. Reasoning Over Structured Knowledge

It should not be too controversial to suggest that something innate exists that allows basic traversal of knowledge structures, an ability afforded to us by the CDCs we defined in the previous section. But in order to really demonstrate the power of this system to perform higher level reasoning, we need to show that it can match structures based on form, a prerequisite shared by both analogical and deductive reasoning.

A. Templates and Form Matching

Deductive reasoning uses form-based matching when determining whether or not an inference rule applies. To use a standard inference rule as an example, assume that we know all men are mortal. Such a statement can take the following form, with X as a variable ranging over some predefined universe:

$$Man(X) \rightarrow Mortal(X) \tag{3}$$

If given the statement $Man(socrates)$, a reasoner would have to first match the form specified in the antecedent of Equation 3. If a match is made, there should be enough information available to inform us how to transform the input statements to produce a new statement (the inferred statement) in accordance with the form specified in the consequent of Equation 3; that resulting formula is $Mortal(socrates)$.

All of this should be quite familiar to anyone who remembers their first experiences with deductive reasoning. But what happens when we instead start with a slightly different statement:

$$Man(plato) \land Mortal(plato) \tag{4}$$

Given now the statement $Man(socrates)$, it does not follow from deductive reasoning that Socrates is mortal. If it does follow from these statements, it is through *analogical* reasoning—Plato was also a man, therefore by analogy it is plausible that Socrates is also mortal. In a template such as that in Equation 3, the antecedent clearly specifies a predicate portion that must be matched exactly (Man), and an object portion that can be anything over which the X variable ranges. In the case of Equation 4, the statements $Man(plato)$ and $Man(socrates)$ do not line up exactly—the objects $plato$ and $socrates$ do share the primary similarity specified by the predicate (they are both men), but an analogical reasoner would likely find similarities between them in other respects: they are both philosophers, they are both from Ancient Greece, etc.

These examples suggest that when matching structured knowledge forms with the end goal of performing deductive or analogical reasoning, at least two things should be available: Firstly, we need to know what constitutes an acceptable match. This may require an exact alignment as in Equation 3, or it may allow a relaxed requirement of surface similarity, as in Equation 4. Secondly, once the match is made, we need to know what resulting inference, or transformation of the input, can be made, and how to do it. This is specified nicely by the consequent portions of both Equations 3 and 4.

To achieve these goals, we introduce the *Template Form* (TF), which builds on the method of representing structured knowledge we defined earlier in this paper to both specify what constitutes an acceptable form match and how to transform the input when such a match is found. Two features are introduced. The first is the Template Chunk, which is a chunk type used by chunks specifically designated to identify complete templates.[2] Template chunks are connected to the template's individual chunks using the template CDC, which is identified in our diagrams by a star encasing the letter 'T'.

For every chunk c in some template identified by the template chunk tc, an associative rule connects these chunks to the template CDC T:

$$(tc, T) \Rightarrow c$$

Since every AR can be weighted as well (not to be confused with the weight of the individual condition chunks within the AR itself), the weights of all ARs outgoing from any particular template chunk adds up to one; this allows us to specify how much matching some particular chunk contributes to the match score of the overall template. Such

[2]We do not in this paper discuss how such templates arise in the first place; this is the subject of future work.

ARs can have weights of zero, and chunks within templates which have zero weights in their corresponding ARs are pictured using a circle with a double border (Figure 2).

Chunks can exist in templates that have zero semantic content. These are called "blank chunks," and will be used when matching templates to other structures.

Our TF method does not allow chunks to have multiple parents. When a quantified variable appears in multiple locations, it is important to preserve the fact that although separate chunks are created for each instance of the variable, since they correspond to the same variable, any chunks matched to these instances must correspond to the same object (or as we will see, this restriction can be relaxed to allow for objects whose chunks have an extremely high similarity). This restriction is reflected in TF using *identity links*, which are pictured using double lines between chunks (Figure 2). Identity links are implemented using the Linker CDC (L, not pictured in Figure 2). such that for any two chunks c_1 and c_2 which are linked, the following ARs are created:

$$(L, c_1) \Rightarrow c_2$$
$$(L, c_2) \Rightarrow c_1$$

B. Matching Structures to Templates

Given some template, actually finding a match to that template is a nontrivial algorithmic problem. In order to avoid some issues that have been raised by the Tailorability Concern [8], an algorithm must be designed that must work with extremely large data sets. With this in mind, the algorithm we chose is designed to be localist—not in the sense of localist concepts we discussed earlier, but rather as the opposite of global, meaning that the algorithm is given a set of chunks as input, and the algorithm only searches chunks in the vicinity of the given chunks. A globally optimal solution is not needed, or even necessarily desirable, in a project which strives primarily for psychological and neurobiological plausibility.

That being said, it would seem that a neurobiologically plausible algorithm would take advantage of the massive parallelism of the brain. For this reason, we explored the use of an Ant Colony Optimization (ACO) algorithm based on [11]. ACO algorithms are examples of metaheuristics, which are used in hard computational optimization problems such as these when a "good enough" solution is needed [3]. Because of space we have to be strictly high-level in our description of this algorithm.

The algorithm is given a a set of target chunks TC and a completed template TMP, which consists of a template chunk, a set of chunks placed into knowledge structures, and all the connecting ARs. (We hope to have future publications address the issue of how such completed templates arise in the first place and are modified over time, but we do not address this in this paper.) The form of this input is notably different from most other models of analogy, which often take predefined source and target structures that are already complete. Instead of a source structure we have a template, and instead of a target structure we simply have a collection of target chunks which may or may not already be structured. The target chunks are ideally a reflective sample of which concepts are currently active in the reasoner's mind.

The algorithm consists of four main routines. The first routine recruits chunks to fill out the target. The second organizes the chunks in the target and template. Next, the third routine actually performs the mapping using an ACO algorithm. Finally, some predicates may be transferred on to the target chunks. We will now describe each part in turn.

1) Recruiting of Target Chunks: The provided target chunks are supposed to be representative of the chunks that have the highest activation levels at some given moment in the NACS. This can be interpreted as being the concepts in the *foreground* of the reasoner's mind. This differs from most models of analogy which come with fully structured target analogs as input. Needless to say, it is possible that the chunks provided to our algorithm as input are insufficient to draw a proper mapping to the provided template, and so the first part of the algorithm attempts to fill out the target somewhat by calling closely associated chunks from memory.

Currently it does this in a very straightforward way: for all chunks $t \in TC$, add all chunks to TC which are identity linked to t. Then add all parents of t. Repeat this process with any recently added chunks until no more can be added (the number of levels of parents is limited to 3 or more, depending on the database). We might also want to add chunks which have a high chunk similarity to t, but that is not done here.

2) Organization of Chunks: Now that we have completed template and target structures, we organize the chunks in TMP and TC into levels, such that all chunks are at the highest levels possible without being on the same level or on a higher level than their parent chunks. The bottom levels are considered to be the 'object levels', and the mapping will be made with the assumption that the two object levels will be mapped to each other, and the same for each level above that.

3) Mapping: An ACO algorithm is used to find a mapping between the chunks. We first start by drawing temporary 'eligibility' links between chunks. For each pair of levels starting from the object levels, an eligibility link is drawn between every pair of chunks (c_1, c_2) such that c_1 is in the template's object level, c_2 is in the target's object level, and the similarity level between c_1 and c_2 is at least 0.8. Blank chunks automatically have eligibility links drawn to every chunk in the corresponding level of the target. Every ant will start with a copy of this list of eligibility links and, as they decide which of these links to add to their mapping, will remove some of these eligibility links from their own copies.

After the pheromone levels are decayed, each ant starts at the object level and selects pairs of chunks from the eligibility links probabilistically, based on several heuristics that either directly or indirectly increase the total match score, again following Sammoud et al. (2005):

- (Lookahead criteria) Does the candidate pair have par-

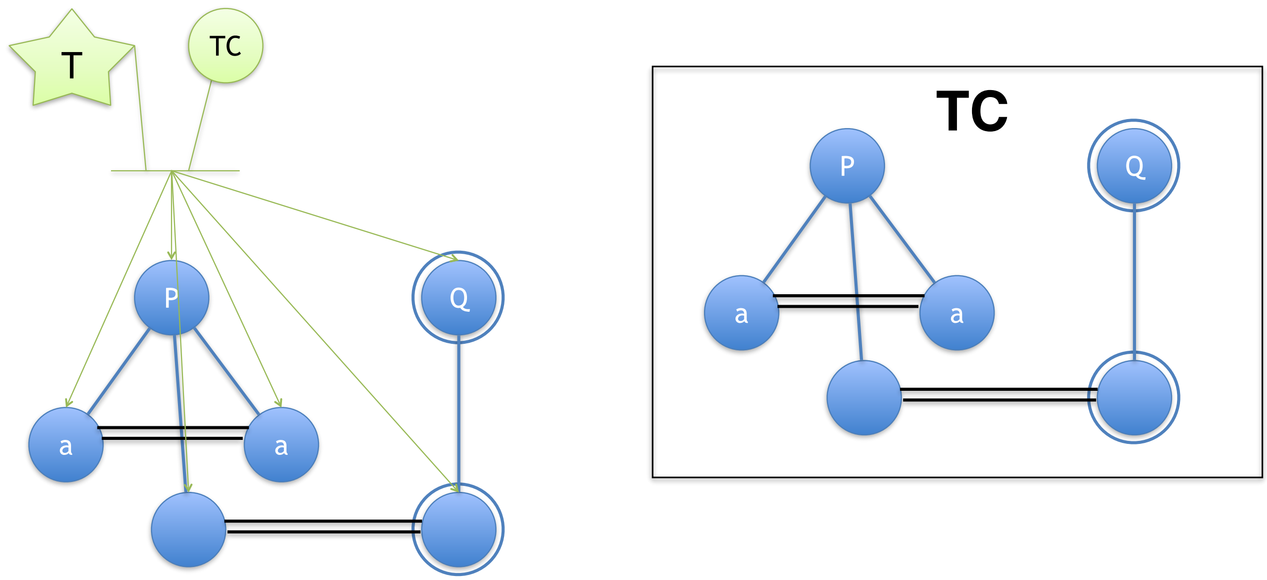

Fig. 2. A typical template with zero-weighted chunks and blank chunks. Simplified version is on the right, which is equivalent to the left picture. Also note that whenever ARs are pictured with multiple heads like in this figure, each head corresponds to a separate AR which has the same tail connections as the others.

ents which are in the eligibility links?

- (Score contribution criteria) Do they have children that are already paired? Do the ARs connecting them to these children use the same CDCs?
- (Pheromone) Check the pheromone attached to this choice, ONLY if this is the very first choice being made by this ant.

With every choice that is made, eligibility links on the same and higher levels may no longer be valid, and so they are temporarily removed before the next choice is made. At the end of each group of ants, the ant with the best match score (which is a function of the number of pairs in the mapping) is compared to the current best score. If the ant's score is better, then pheromone is deposited on each pair in that ant's mapping.

Each group of ants and a single deposit of pheromone constitutes a single iteration. After a certain number of iterations, the best mapping is returned.

4) Transfer: The best mapping score (s) is then divided by the theoretical maximum score (s_{max}). If this amount is greater than a certain tolerance t (usually 0.8), then a bottom-up search is made for chunks in the template that were not mapped to anything. If that chunk's weight within the template w is such that $\frac{s-w}{s_{max}} \geq t$, then a copy of the chunk is made and can be transferred to the target, and any necessary CDC-related ARs are created. s is set to $s - w$, and the process is repeated.

C. Performing Deductive and Analogical Reasoning

We are now in a position to demonstrate how the knowledge representation style and algorithm we describe can perform deductive and analogical reasoning. We will demonstrate by using the two examples presented above, Equations 3 and 4.

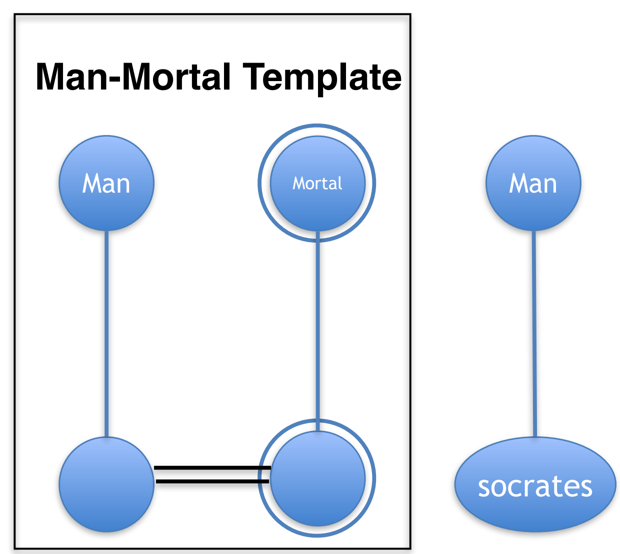

Fig. 3. Template and target used for the deductive reasoning example in Formula 3.

As can be seen in Figure 3, deductive reasoning is performed by having templates with blank chunks for quantified variables, and zero-weighted chunks for the consequent chunks. This way if the antecedent (the non-zero-weighted chunks) is matched to a high enough tolerance, then the

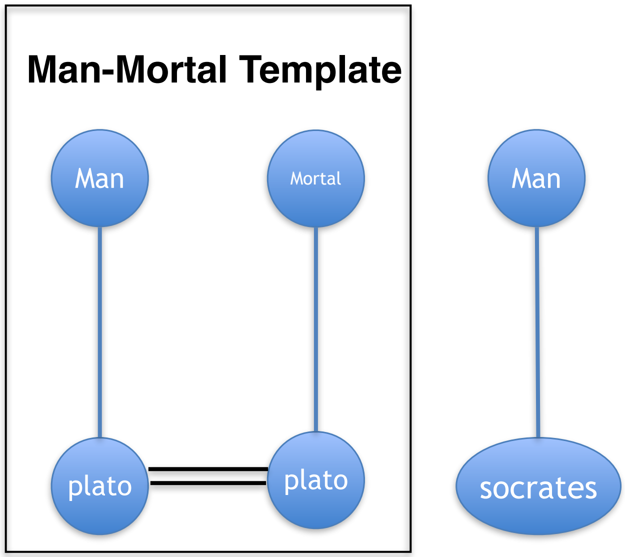

Fig. 4. Template and target used for the analogical reasoning example in Formula 4.

consequent (the zero-weighted chunks) are automatically created, representing an inference.

Analogical reasoning, on the other hand, may not have a finalized template. An extra step is required that first collects source chunks using a similar process to the *"Recruiting of Target Chunks"* step described above, and then tries (in parallel) different transformations of the source chunks into templates. The algorithm used in deductive reasoning can then be used. We hope to expand on this further in a later paper.

V. FOL Expressivity

The work described in this paper was designed in part to be at least as expressive as first-order logic (FOL), the touchstone for assessing the expressivity of extensional logics [4].[3] Many extensional logics are even more expressive than FOL, but moving beyond FOL means sacrificing desirable meta-properties; second-order logic, for instance, while strikingly convenient computationally, is provably incomplete. In this section we show that a major part of the goal to at least reach FOL has been met. We do this by first showing how the full syntax of FOL can be represented in our knowledge structures.[4]

[3]Extensional logics do not permit non-extensional operators like *Believes*, which e.g. can be applied to propositions ϕ whose semantic values do not predictably generate a semantic value for the "outer" proposition. E.g., even if we know that ϕ is false, we cannot infer that 'Jones believes ϕ' is false (or, for that matter, that it's true). For introductory discussion of this phenomenon in the context of computational cognitive modeling, see [2]. Future work will seek to enable, in CLARION, intensional reasoning, which seems to be part and parcel of human moral reasoning, which we ultimately seek to model (see §VI).

[4]We spare the reader a straightforward proof by induction over terms and formulae in first-order logic, in which one demonstrates that every formula in a countably infinite progression of all first-order formulae can be represented in the scheme we have introduced.

In order to represent FOL formulae, we adopt a structure that directly maps to human-readable syntax. We represent the universal and existential quantifiers as if they were higher-level predicates, using identity links to connect quantified variables to instantiations within the variable's scope. This can be seen in Figure 5, along with an example of negation, which is similarly treated as a sort of single-place predicate.

Such structures can exist in this format for easy recall by the reasoner. When they are to be used in active reasoning processes, however, they may be temporarily transformed (e.g., to the structure in Figure 3), perhaps by some process which originates in CLARION's Action-Centered Subsystem (ACS). This process is a bit more involved than the examples we demonstrate here. For example, this process needs to distinguish between the chunks corresponding to the universal and existential quantifiers in order to treat their corresponding structures differently in inferences.

Allowing for a native representation that contains quantifiers is a first step in simulating a so-called "natural" reasoning process, that is, a set of mechanisms that are known to better correpsond to how humans, as opposed to machines, reason (e.g., something akin to *natural deduction*, introduced in [7]). Aiming at natural reasoning may seem an odd choice, considering that modern automated theorem provers tend to prefer methods such as first-order resolution, but we remind the reader that our goal here is to model reasoning in a psychologically plausible way — and in a way that integrates with sub-symbolic processing in the human system.

VI. Conclusion

In this paper, we presented a method of representing structured knowledge in CLARION that is psychologically plausible and satisfies desiderata **D1**–**D4**. This allows us to reason over these structures using a psychologically plausible algorithm which has the nice property of conservatively performing both analogical and deductive reasoning using mostly the same underlying mechanisms—those of template matching. We also defined an algorithm that performs a localist search designed to be used with extremely large, but connected databases, in the spirit of the recommendations made by [8].

A primary goal of this larger project is to perform ADMR—Analogico-Deductive *Moral* Reasoning—in CLARION. The work described here will be used to create a system that can not only use ADR [9] to reason about moral and ethical situations, but will have conclusions whose justifications are traceable and can be verified independently.

References

[1] Selmer Bringsjord. Is the Connectionist-Logicist Clash One of AI's Wonderful Red Herrings? *Journal of Experimental and Theoretical AI*, 3.4:319–349, 1991.

[2] Selmer Bringsjord. Declarative/Logic-Based Cognitive Modeling. In Ron Sun, editor, *The Handbook of Computational Psychology*, pages 127–169. Cambridge University Press, Cambridge, UK, 2008.

[3] Marco Dorigo and Christian Blum. Ant Colony Optimization Theory: A Survey. *Theoretical Computer Science*, 344(2-3):243–278, 2005.

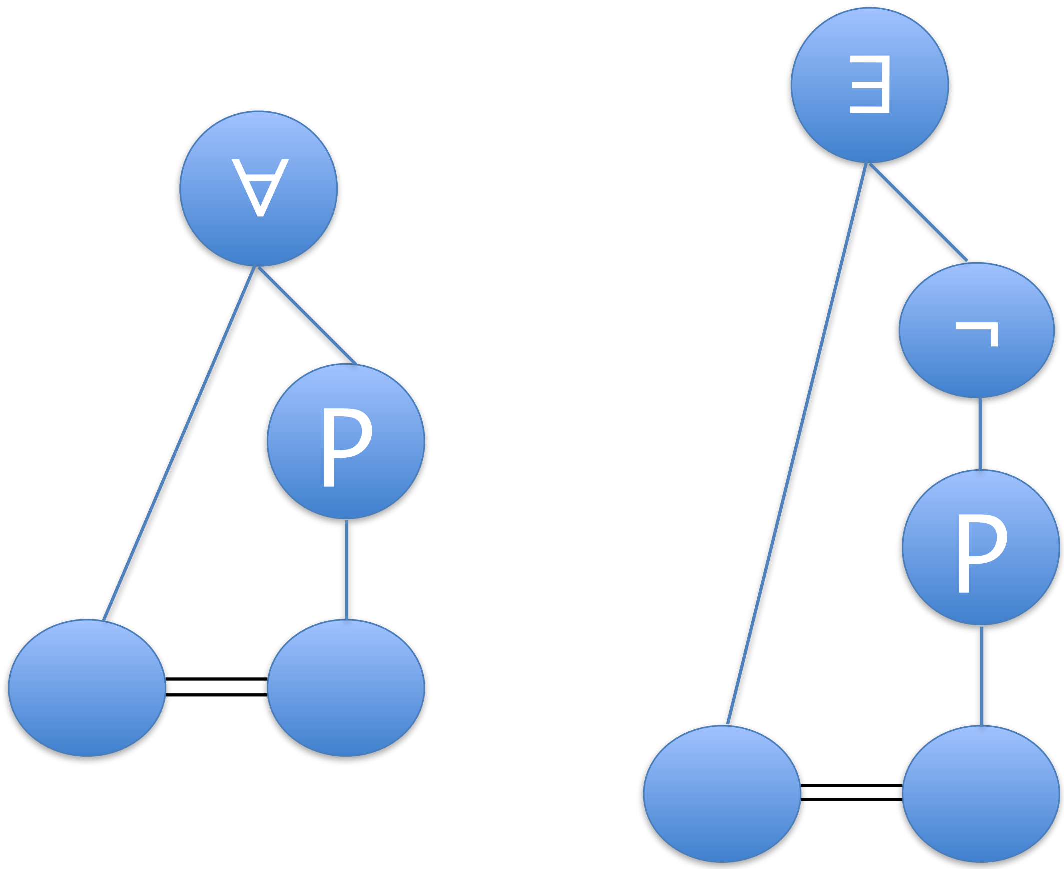

Fig. 5. First order logic formulas $\forall_x P(x)$ and $\exists_x \neg P(x)$.

[4] H. D. Ebbinghaus, J. Flum, and W. Thomas. *Mathematical Logic (second edition)*. Springer-Verlag, New York, NY, 1994.

[5] Geoffrey E. Hinton, J.L. McClelland, and David E. Rumelhart. Distributed Representations. In David E. Rumelhart and J.L. McClelland, editors, *Parallel distributed processing: explorations in the microstructure of cognition*, volume 1. MIT Press, 1986.

[6] John E. Hummel and Keith J. Holyoak. Relational Reasoning in a Neurally-plausible Cognitive Architecture: An Overview of the LISA Project. *Cognitive Studies: Bulletin of the Japanese Cognitive Science Society*, 10:58–75, 2003.

[7] Stanislaw Jaśkowski. On the Rules of Suppositions in Formal Logic. *Studia Logica*, 1:5–32, 1934.

[8] John Licato, Selmer Bringsjord, and Naveen S. Govindarajulu. How Models of Creativity and Analogy Need to Answer the Tailorability Concern. In Tarek Richard Besold, Kai-uwe Kühnberger, Marco Schorlemmer, and Alan Smaill, editors, *Proceedings of the IJCAI 2013 Workshop on Computational Creativity, Concept Invention, and General Intelligence*, Beijing, China, 2013.

[9] John Licato, Selmer Bringsjord, and John E. Hummel. Exploring the Role of Analogico-Deductive Reasoning in the Balance-Beam Task. In *Rethinking Cognitive Development: Proceedings of the 42nd Annual Meeting of the Jean Piaget Society*, Toronto, Canada, 2012.

[10] John Licato, Naveen S. Govindarajulu, Selmer Bringsjord, Michael Pomeranz, and Logan Gittelson. Analogico-Deductive Generation of Gödel's First Incompleteness Theorem from the Liar Paradox. *Proceedings of the 23rd Annual International Joint Conference on Artificial Intelligence (IJCAI-13)*, 2013.

[11] Olfa Sammoud, Christine Solnon, and Khaled Ghédira. An Ant Algorithm for the Graph Matching Problem. In *5th European Conference on Evolutionary Computation in Combinatorial Optimization (EvoCOP 2005)*. Springer, 2005.

[12] Keith E. Stanovich. *Who is Rational? Studies of Individual Differences in Reasoning*. Lawrence Elrbaum Associates., Mahwah, NJ, 1999.

[13] Ron Sun. A Microfeature Based Approach Toward Metaphor Interpretation. In *Proceedings of the International Conference on Artificial Intelligence (IJCAI-95)*, pages 424–430. Morgan Kaufmann, 1995.

[14] Ron Sun. Robust Reasoning: Integrating Rule-Based and Similarity-Based Reasoning. *Artificial Intelligence*, 75(2), 1995.

[15] Ron Sun. *Duality of the Mind: A Bottom Up Approach Toward Cognition*. Lawrence Erlbaum Associates, Mah- wah, NJ, 2002.

[16] Ron Sun and Xi Zhang. Accounting for Similarity-Based Reasoning within a Cognitive Architecture. In *Proceedings of the 26th Annual Conference of the Cognitive Science Society*. Lawrence Erlbaum Associates, 2004.

[17] Ron Sun and Xi Zhang. Accounting for a Variety of Reasoning Data Within a Cognitive Architecture. *Journal of Experimental and Theoretical Artificial Intelligence*, 18(2), 2006.

[18] A. Tversky. Features of Similarity. *Psychological Review*, 84(4):327–352, 1977.