Building Analogy Systems: Some Lessons Learned

Kenneth D. Forbus FORBUS@NORTHWESTERN.EDU
Thomas Hinrichs HINRICHS@NORTHWESTERN.EDU
Irina Rabkina IRABKINA@U.NORTHWESTERN.EDU

Qualitative Reasoning Group, Northwestern University, 2133 Sheridan Road, Evanston, IL, 60201 USA

Abstract

Analogy is a powerful method of reasoning and learning, but it does not stand alone. Analogical reasoning, we have found, works best when tightly integrated with other forms of reasoning and learning. This paper summarizes some lessons learned in creating reasoning and learning systems that incorporate analogy for a variety of tasks. The key lessons are: (1) a general-purpose, expressive ontology is very useful, (2) reified contexts for reasoning, called microtheories, provide a powerful way to organize knowledge, including cases, (3) the requirements of analogical retrieval and generalization provide interesting constraints on knowledge base design. These lessons are illustrated with examples from prior cognitive systems projects, and three open questions are discussed.

1. Introduction

Analogy is a powerful method of reasoning and learning for cognitive systems. But analogy by itself is not enough to build cognitive systems. Cases must be constructed, from sensor information (Chen & Forbus, 2018), from natural language understanding (Barbella & Forbus, 2011, 2016; Blass & Forbus, 2016), by reasoning (Blass & Forbus, 2015), and/or by combining relevant general knowledge (Mostek et al. 2000). The inferences drawn via analogy must be tested, to see if they are reasonable. This can involve logical deduction (e.g. Klenk & Forbus, 2009), visual processing (e.g. Lovett & Forbus, 2017), or game playing (e.g. Hinrichs & Forbus, 2011). Therefore it makes sense to tightly integrate analogy with other forms of reasoning and learning.

This paper describes some lessons we have learned in creating reasoning and learning systems in the Companion cognitive architecture (Forbus et al 2009; Forbus & Hinrichs 2018). In Companions, analogical matching, retrieval, and generalization are central, but operate in concert with logical reasoning, spatial reasoning, planning, and natural language understanding. Specifically, we have found that

1. A broad coverage, highly expressive ontology is extremely useful. As described below, we use the Cyc ontology, although not the Cyc software.
2. Reified contexts for reasoning, called microtheories, provide a powerful way to organize knowledge. Cases can be implemented as microtheories, which simplifies integrating analogy with other forms of reasoning.
3. Analogical retrieval and generalization place strong constraints on knowledge base design and will become increasingly important as the field scales up to human-sized knowledge bases.

While the work described here uses our structure-mapping models of analogy (summarized below), we believe that these lessons are potentially useful for any cognitive system that employs analogy (e.g. Besold et al. 2015; Fitzgerald et al. 2016; Homes & Winston, 2016, 2017; Vattam & Swaroop 2012) or could do so in the future (e.g. Dannenhauer & Munoz-Avila, 2013; Laird 2012; McShane et al. 2012).

2. Background

Our approach to analogy is based on Gentner's (1983) structure-mapping theory, which views analogy and similarity as a mapping between structured relational representations. Object attributes (modeled as unary predicates) provide type information and concrete perceptual information, while relations encode both relations between entities (e.g. `above`) and relations between other relations (e.g. `implies`, `cause`). Mappings follow a set of principles that have been successfully tested against psychological evidence across a variety of tasks and domains. The Structure-Mapping Engine, SME (Forbus et al. 2017b), implements the matching operation of structure-mapping theory. SME takes as input two structured, relational cases and computes up to three mappings between them. Each mapping consists of a set of *correspondences* that align entities and statements in one description with the other. Mappings also have a numerical *similarity score*, providing an overall estimate of the structural quality of the match. Mappings also have *candidate inferences*, which can project relations, attributes, and entities from the base to the target, or the target to the base, using the correspondences. These candidate inferences serve two purposes in analogical reasoning and learning. 1) They provide a source of conjectures about the descriptions which can be used in reasoning. These are not guaranteed to be valid (unless the relational structure projected is itself logically valid), but provide grist for abductive and inductive reasoning. 2) Candidate inferences also provide a representation of psychologically salient differences, called *alignable differences*. Alignable differences are useful in diagnosis. SME operates in polynomial time, using a greedy merge algorithm.

There are many other models of analogical matching. Some work on finding a common abstraction (e.g. HDTP, Weller & Schmid, 2006), others are attempting to model neural implementations (e.g. LISA, Hummel et al. 2014). We note that SME's middle-out algorithm captures time-course processing found in metaphor interpretation that other models do not (Wolff & Gentner, 2011), and has been used with a wider range of examples, as well as more complex examples, and with multiple other forms of reasoning, as noted above, compared to other models.

A second psychological process in structure-mapping theory is *analogical retrieval*. In case-based reasoning, retrieval is often tailored using predictive features of a domain or task (Riesbeck & Schank, 1989). By contrast, human retrieval of experiences seems to rely on the same process (or multiple processes with the same overall properties) across a variety of domains. Human retrieval is sensitive to surface properties, for example, which is reasonable because things that look alike often are alike. As knowledge grows in an area, retrieval starts to reflect more abstract knowledge as well, e.g. experts are more likely to retrieve examples that follow the same principle as the current situation even if they are from a different domain (Catrambone, 2002). Our model of analogical retrieval is MAC/FAC, which stands for "Many Are Called/Few Are Chosen" (Forbus et al. 1995). MAC/FAC operates over a case library, whose contents consist of cases, which are structured descriptions as described above. Given a new structured case, the *probe*, it returns up to three of the most similar cases in the case library. It does this by using a two stage process, each of which can be conceptually viewed as map/reduce. In the first stage, MAC, a dot product between a content vector for the probe and for every case in the library is computed. Content vectors are automatically computed from structured cases, with each position in the vector indicating the relative frequency of statements using a predicate appear in that description. The dot product of two content vectors provides an estimate of SME's numerical similarity scores for the original structured cases. The cases corresponding to the top dot product (or up to three if they are very close) are passed to the next stage, FAC. FAC uses SME in parallel to compare each case to the probe, and again returns the best (or up to all three, if they are very close) as the reminding. Notice that, unlike traditional case-based reasoning systems, this is not an indexed memory system.

Structure-mapping theory also argues that generalization occurs by repeated comparisons, keeping the overlapping parts and allowing non-overlapping parts of the description to fade as more examples are accumulated. This is implemented in the Sequential Analogical Generalization Engine, SAGE (McLure et al. 2015). The analogical model of a concept is represented by a *generalization pool*, which accumulates examples and generalizations incrementally. For example, in word learning, a generalization pool would be used for each word (e.g. Lockwood et al. 2008; McFate & Forbus 2016). Given a new example of a concept, SAGE uses MAC/FAC to retrieve the closest generalization or example. If the degree of similarity is over a threshold, the retrieved item and the new example are assimilated. Assimilation for two examples consists of creating a new generalization, replacing non-identical entities with blank entities (not logical variables) and adding probabilities for each statement based on whether or not they are overlapping. That is, aligned statements will have a probability of 1.0, while statements not in the mapping will have 0.5. If the retrieved item is a generalization, the probabilities and generalized entities are updated. When the probability of a statement drops too low, it is culled from the generalization. Notice that this process supports handling disjunctive concepts, since multiple generalizations can be formed. It also handles outliers, in that it maintains all un-generalized examples. SAGE has also been extended to automatically find and use near-misses (Winston, 1970) to improve discriminability of its models (McLure et al. 2015). We have also extended this method of analogical generalization to use temporary generalization pools in working memory, to handle *interim generalizations* (Kandaswamy et al. 2014), which support within-problem learning. Working memory generalization pools are tightly limited in the number of items they can store (the current default is five). Retrieval is based on using SME serially over the contents of the pool, organized by recency. If any match is over the assimilation threshold, that pair leads to a new (or updated) generalization which then becomes the most recent item. Otherwise, the new example becomes the most recent item.

3. Expressive Ontologies Support Analogy

Many analogy systems have only been used with small ontologies that have been hand-generated for specific examples (e.g. Gust et al. 2004). Our alternative is to use an off-the-shelf independently-developed general purpose ontology which provides a well-developed and extensively tested representational vocabulary. A general ontology provides a common representational target language for natural language understanding, sketch understanding and vision, such that they can all produce cases or contribute to a common multimodal case representation from natural interaction. This enables non-experts to generate cases (e.g. Forbus et al. 2018) and reduces the temptation for experimenters to hand-craft representations to particular domains or tasks and to introduce unintentional bias. Moreover, while lexicons can support language processing, they do not by themselves have the inferential connections needed to support reasoning making their use in analogy limited (e.g. Veal 2006). Our ontology incorporates an extensive lexicon and mapping to concepts and predicates.

We have found the Cyc ontology to be extremely useful in our research. Many have strong negative opinions of Cyc. Domingos (2015), for example, calls Cyc "the most notorious failure in the history of AI" but provides no evidence for this assertion. Cycorp itself publishes little, but from their papers (e.g. Pierce et al. 2012) and versions of their software made available to researchers, it is easy to see quite a different picture. While their reasoning engine is extremely powerful, we have an alternative approach (i.e. making analogy central) and prefer full source code access. So we extract the Cyc ontology from their system and use it in our own reasoning engines. These representations, originally from ResearchCyc, but now from OpenCyc, have been fueling our work for the last 20 years.

Part of the advantage of using the Cyc ontology is scale. The version we use currently has 87,206 concepts, 26,557 relations, and 5,302 functions. These are mutually constrained by 1,353,564 facts. This is a heavily pruned subset of the original OpenCyc, but also includes additions our own group has made. These extensions include a broad natural language lexicon, frame semantics drawn from FrameNet (Fillmore et al. 2002) for natural language processing, an implementation of qualitative process theory (Forbus, 1984), and visual representations used for sketch understanding (Forbus et al. 2017a) and for visual reasoning and learning (Chang 2016; Chen & Forbus, 2018).

A key part of our ontological extensions include concepts, predicates, and relations that reify concepts and options of analogy (Forbus et al. 2002). Matching, retrieval, and generalization are all accomplished by queries and commands expressed using the ontology. For example, terms constructed via functions (also known as *non-atomic terms*, or NATs) are used to denote a match, that is, a comparison between a base and target. A match can have several mappings, each represented in the declarative part of the system by terms. These terms are constructed on demand, based on queries. Thus a query might involve retrieving a case, and examining its candidate inferences for particular kinds of conjectures, all expressed via declarative rules.

Here is an example from Sketch Worksheets, a deployed system that uses SME to provide advice to students about their sketches (Forbus et al. 2017a; Forbus et al. 2018). Sketch Worksheets are implemented in CogSketch, which performs visual and spatial reasoning with digital ink. This includes the construction of qualitative and quantitative visual relationships. Advice is generated for students by using SME to compare their sketch with an instructor's sketch. Here are the subgoals which are executed, sequentially, to make this comparison:

```
(sketchFor ?target-subsketch ?sketch) ;; Find the overall sketch
;; Construct the base and target cases from the appropriate subsketches
(classicWorksheetCaseConstructor ?base-subsketch ?base-case)
(classicWorksheetCaseConstructor ?target-subsketch ?target-case)
(numAnswers 1 ;; Compute within-domain match constraints
        (classicWorksheetMatchConstraints
          ?base-subsketch ?target-subsketch ?constraints)
;; Students can ask for help multiple times, erase computations from prior requests if any
(tell (retractAllAnalogyResults ?sketch))
;; Perform the analogy, binding a term for the SME instance to ?initial-match
(matchBetween ?base-case ?target-case ?constraints
              ?initial-match)
;; Use the qualitative match to frame quantitative matching, if included in solution
(quantitativelyConstrainedMatch ?initial-match ?match)
```

The `?base-subsketch` is the teacher's solution, the `?target-subsketch` is the student's drawing. These subsketches consist of digital ink and qualitative plus quantitative relationships automatically computed from that ink. Some of these relations, expressed in English via natural language generation in CogSketch's authoring environment, are marked by the instructor as important. The predicate calculus representations of cases are computed from the sketches, in ways that ensure the visual analyses required by what the instructor marked as important are automatically performed (which is what `classicWorksheetCaseConstructor` does). The `tell` subgoal causes results from any prior request for feedback to be retracted, since the student's sketch may now be quite different. The subgoal `matchBetween` calls SME on the base, target, and match constraints, binding `?match` to a non-atomic term denoting the match. The match constraints are automatically computed, i.e. that type constraints on entities should be respected – these are within-domain analogies! Some sketch worksheets have quantitative constraints that must also be respected, and the qualitative match is used to frame the checking of these quantitative constraints, if any, in the last subgoal (Chang & Forbus, 2012).

Integration is not without its subtleties. When integrating structure-mapping with Cyc-style representations, one such subtlety is determining whether a fact should be represented as an attribute or a relation. Attributes are unary predicates that describe properties, such as

`(Cat Nero)` & `(BlackColored Nero)`

In Cyc, concepts such as `Cat` and `BlackColored` are collections, and these statements would be written as

`(isa Nero Cat)` & `(isa Nero BlackColored)`

The `isa` form is preferred for rule-based reasoning because it's easier to detect when a rule is relevant. (Triggers of the form `(?c ?o)` can complicate rule indexing.) But for structure-mapping, the attribute form is superior for two reasons. First, in the `isa` form, concepts are treated as entities. That means, for example, that they are not used in the content vectors that drive the MAC stage of MAC/FAC, which reduces discrimination and sensitivity to surface properties. The second reason is that the `isa` format gives rise to many more match hypotheses in SME's middle-out algorithm. Consider two descriptions of single objects, each of which is described by the same N distinct attributes (e.g. another black cat, as per above). With the attribute formulation, there will be exactly N hypothesized correspondences constructed during matching, one for each identical attribute, and only a single mapping. With the `isa` formulation, there will be N^2 hypothesized correspondences constructed, because each `isa` statement can match every other `isa` statement. There will also be multiple distinct mappings, corresponding to different attributes aligning with others (but at most only three will be generated, a constraint built into SME). The fact that the collections are different, in this representation, is irrelevant – non-identical entities can match freely under structure-mapping. In other words, treating attributes as entities in the `isa` formulation does not respect their importance as properties in structure-mapping.

This transformation is more general than the `isa` relationship. For example, in connecting up a reasoning system to a simulation, it is common to use binary predicates to dynamically reify properties of the simulated world for reasoning. One of the domains we use with Companions is Freeciv, an open-source version of Civilization 2. Consider the following statements about a situation in Freeciv:

```
(unitType Unit-107 FC-Unit-Workers)
(terrainAt (FreecivLocationFn 61 41) FC-Terrain-Plains)
(fcObjectAt Unit-107 (FreecivLocationFn 61 41))
```

The first two statements are best viewed as attributes because `FC-Unit-Workers` and `FC-Terrain-Plains` are types. On the other hand, the last statement is best viewed as a binary relation, because the tile 61,41 is itself an entity, as is `Unit-107`. For building cases to learn about the effects of actions, such statements are automatically translated into attributes by using a pre-existing higher-order function in Cyc, which denotes a subconcept of a broader concept, here, units that are workers and terrain that are plains:

```
((SubcollectionOfWithRelationToFn
     FreeCiv-Unit unitType FC-Unit-Workers) Unit-107)
((SubcollectionOfWithRelationToFn
     FreecivLocation terrainAt FC-Terrain-Plains)
         (FreecivLocationFn 61 41))
(fcObjectAt Unit-107 (FreecivLocationFn 61 41))
```

This ability to compose new attributes on demand illustrates the utility of having an expressive ontology. We are not aware of any other broad ontology that includes such higher-order constructs.

4. Microtheories as Cases

Human knowledge is rich and complex. We are capable of holding multiple perspectives in mind at once. We know that Sherlock Holmes is a fictional character. But we also know that he lives in 221b Baker Street. Moreover, we can juggle multiple overlapping fictional worlds: In the original stories and in the BBC Sherlock series, he lives in London, while in the Elementary series, Holmes lives in Manhattan. Multiple perspectives also arise when considering alternatives, in diagnosis or design or analysis. Dealing with complex phenomena often requires using multiple models at different levels of granularity. For example, Newtonian, relativistic and quantum mechanics all hold sway at different levels of size and speed. In other words, human knowledge is contextual and compartmentalized.

The Cyc ontology handles this via *microtheories* (Guha 1991). A microtheory can be thought of as a bin that contains facts. For example, the `WorldGeographyMt` microtheory contains a large number of facts about geography. On the other hand, the `OrdinalReasoningMt` contains Horn clause rules that support drawing inferences about ordinal relations. Microtheories are connected to each other via an inheritance relation, `genlMt`. If reasoning about the relative sizes of two countries, both of these microtheories might be needed, and can be made accessible by having the microtheory where the reasoning is occurring inherit from them both. Reasoning is performed with respect to a *logical environment* – a microtheory plus all of the microtheories that it inherits from, recursively. Since `genlMt` statements can be asserted and retracted, reasoning systems can dynamically reconfigure which aspects of their knowledge they bring to bear on a problem. Our reasoning engine is organized so that both inheritance and dynamic reconfiguration of logical environments is extremely fast, since these operations tend to be heavily used.

For analogical reasoning, microtheories provide a natural implementation mechanism for cases. That is, cases simply are microtheories. For example, in qualitative reasoning in the Cyc ontology, qualitative states are implemented via microtheories. That means that they can be compared by simply using those microtheories as the base and target for SME.

In our cognitive modeling work, we have represented events, stories, grammatical forms, and even memories as microtheories that we then used as cases for analogy. For example, when modeling a training study in which children improved their theory of mind reasoning abilities after hearing three structurally similar stories (the last of which contained a surprising twist), each story was represented as a microtheory (Rabkina et al., 2017). Each story microtheory contained predicate calculus representations of the interaction between the child and the experimenter in the original study. Just as the children heard the stories in order, the cases were presented to our model (Analogical Theory of Mind; AToM) one after the other. Each was analogically compared with previous cases and, where appropriate, generalized. Importantly, the surprising story triggered a search in long term memory which resulted in the retrieval of a memory—itself represented in a microtheory. In a subsequent study, we proposed a mechanism by which children can bootstrap theory of mind from language, via AToM (Rabkina, McFate & Forbus, 2018). Bootstrapping, in the model, relies in part on transferring the structure of the grammatical form of a sentence onto its arguments via candidate inference. Both the grammatical form and the argument contents were represented as microtheories and compared via SME to obtain the requisite candidate inference.

To be sure, not all microtheories should be treated as cases: `UniversalVocabularyMt` (at 508,195 facts) and `BaseKB` (at 211,404 facts) are far too large to participate in comparisons. Even when cases are constructed by reasoning, not all of the facts used in reasoning are relevant for comparisons – reasoning processes often require intermediate, bookkeeping facts in order to reach desired conclusions. So analogy control predicates (Forbus et al. 2017b) are used to filter out such facts. For example, in using SME for generating tutoring advice in Sketch Worksheets as mentioned above, about 30 types of statements are filtered out automatically. These analogy control predicates can be grouped into different microtheories and swapped in and out of the current reasoning context as needed.

5. Retrieval, Generalization, and Knowledge Base Design

Case storage and generalization across cases constitute persistent, long-term changes to a system's memory. This means that a knowledge base to support integrated reasoning must also support the case libraries needed for MAC/FAC and the generalization pools needed for SAGE. Since MAC/FAC is used by SAGE, in fact generalization pools are implemented as a subclass of case libraries.

To support analogical operations in a general cognitive architecture, we have found it necessary to organize memory into multiple task or domain-specific case libraries. For example, in learning a complex strategy game like Freeciv, generalization pools are used to build up models of the immediate effects of actions. One pool is created for each action. Similarly, case libraries are used to record cases relevant to a type of decision, including both suggested actions and lessons from negative experiences (Hinrichs & Forbus, 2007).

In our knowledge base, persistent objects are used to store and index case libraries and generalization pools. This turns out to be important for efficiency and scaling because it allows us to use the same low-level database indexing that is used for fact retrieval. Case libraries and generalization pools are each named using non-atomic terms. For example, `(OutcomeExpectationFn doMove)` denotes the generalization pool for computing the expected outcomes of a `doMove` action in Freeciv, so that every execution of that primitive action adds a new case to the pool. Cases are added to case libraries and generalization pools by `tell` commands from the declarative level of the system, and retrievals are performed by queries, whose results are passed back from the underlying procedures by binding variables in the query. Case libraries automatically compute content vectors when a case is added, and they are persisted for efficiency. The history of the specific entities that went into a generalized entity are maintained in generalization pools, to provide grist for further analysis.

Consider a typical classification task, where a set of labeled examples has been fed into SAGE. Each labeled concept is represented by a generalization pool. But to classify a new example, we need to perform analogical retrieval over the entire set of relevant generalization pools. We handle this by allowing case libraries to be hierarchical, i.e. all of the generalization pools are sub-libraries of a containing case library. Given a new example, the generalization pool from which the closest retrieved item came is used to provide the label. This is reminiscent of traditional discrimination-based retrieval, except that the tests are much more sophisticated structural analogies.

For example, analogical word sense disambiguation (Barbella & Forbus, 2013) relies on having a generalization pool for each pair of a word and a sense for that word. Thus there are separate generalization pools for <"star", `FamousHuman`>, <"star", `AstronomicalBody`>, and <"star", `Star-PlaneFigure`>. Each use of "star" that is decoded into one of these meanings is used to construct a case, including the automatically generated lexical, syntactic, and semantic facts connected to that decision, and stored in the appropriate generalization pool. Each pool is building up a model of the circumstances in which that word sense is used. By picking the most similar from among the senses for a word, this provides evidence as to how to interpret new occurrences of that word.

There is ample evidence suggesting that human conceptual structure is hierarchical, and a variant of SAGE has been built which can construct hierarchical concepts (Liang & Forbus, 2014). While promising, this algorithm is batch instead of incremental, which is unsatisfactory given our hypotheses about the ubiquity of analogy in cognitive systems. Real life is not typically divided into train/test phases; the ability to deal with changing distributions of examples and incrementally introduce hierarchical concepts seems extremely important for robustness.

6. Open Questions

While we have learned a lot, there are many open questions. Here are three that we view as particularly important.

First, how contextually sensitive should retrieval and generalization be? On one hand, we currently use analogy control predicates in retrieval and generalization. That is, predicates that are inferred as `notForAnalogy` within the logical environment from which a retrieval query is made are dynamically filtered from content vectors. It could be argued that this goes too far, in that the setup of case libraries and generalization pools is already highly contextualized. But it also may not go far enough: recency matters psychologically, i.e. a case is more likely to be retrieved if it has been retrieved recently. Priming also matters, cases that include concepts associated with recent experiences are also more likely to be retrieved. MAC/FAC currently does not address either of these factors.

Second, as the number of case libraries, generalization pools, and cases grows, scaling of retrieval and generalization will be an issue. A recent back of the envelope estimate (Forbus et al. 2017c) suggests that for a lower bound estimate of human experiences, 45 million cases and 2 million generalization pools is not an unreasonable projection. The only part of MAC/FAC and SAGE that depends on the number of cases in a library is the MAC stage of MAC/FAC. Conceptually, MAC can be done in a data parallel manner, but doing that efficiently in today's computing technologies is tricky. A case might have 10-50 non-zero entries in a content vector, out of a potential 119 thousand entries. This level of sparsity means that zero-filling algorithms and sorting algorithms for handling sparse vector operations are very inefficient. Some clever engineering will be needed here.

Third, learning encoding schemes effectively is becoming a central problem for us. An encoding scheme takes information expressed in a natural modality, such as text, sketches, images, video, or other systems, such as a simulation, and automatically produces cases for analogical learning or reasoning. Encoding schemes are complicated because information from natural modalities can be understood from multiple perspectives and multiple levels. Most of our models and tasks use a fixed encoding scheme that has been set up by the experimenters. We want to move to automatic learning of encoding schemes. Some experiments on this have been done already (McLure & Forbus, 2012), but there are several difficulties. First, the ultimate metric is producing robust and effective generalizations that lead to improved performance. While analogical learning is data-efficient, this can still be expensive. Second, how should cases encoded with prior strategies continue to be used, if at all? If data is a continuing stream and experiences are plentiful, flushing them might be a reasonable strategy. But if a system must continually make decisions while learning incrementally (which is more realistic than the train/test cycle commonly used in machine learning), then maintaining multiple encodings and case libraries while evaluating alternate encoding schemes might be a better strategy.

7. Conclusions and Future Work

We have argued that building analogy systems in the 21st century should exploit large-scale knowledge bases. These resources provide many advantages for analogy and cognitive architectures. In working with the Cyc ontology over the last two decades, we have learned several important lessons:

1. A general-purpose broad and expressive ontology provides resources for rapidly developing knowledge in new domains, provides a target for automatic representation construction by language, sketching, vision, and reasoning, and provides a means of tightly integrating analogy with other forms of reasoning.
2. There is a natural alignment between the notion of microtheory in the Cyc ontology and the notion of case in analogy. By treating microtheories as cases, we simplify applying analogy to descriptions created by other forms of reasoning, and in turn using those other forms of reasoning to verify analogical inferences.
3. Case libraries and generalization pools should be viewed as part of the long-term knowledge of a system. They need to be treated as first class entities in knowledge bases. Hierarchical case libraries are useful for building classification systems.

We believe that analogy should become a regularly used capability in cognitive systems, and we hope that this paper will help others in using it. Cognitive systems need rich ontologies and large relational knowledge bases. Fortunately, the rise of large-scale knowledge bases, like Cyc, ResearchCyc, OpenCyc, Freebase, Google's Knowledge Graph, and Microsoft's Satori, are a sea change in the importance of relational processing in AI and machine learning. We should take advantage of this. Analogy research has much to offer, and benefits in turn from being able to use off-the-shelf rich relational representations.

One problem with all of the knowledge bases above, except for OpenCyc and Freebase, is that they are proprietary. While we have the highest regard for Cyc and ResearchCyc, we have moved away from ResearchCyc to OpenCyc because we need to be able to distribute the results of our work to everyone. We are continuing to distribute our own developments building on the OpenCyc ontology. This includes knowledge base contents to support for analogical reasoning, qualitative reasoning, and natural language resources to support research in deep understanding and learning by reading, all available as an open-license resource. We hope that by doing so, this will encourage and facilitate further research in knowledge-rich AI more broadly.

Acknowledgements

The work described in this paper has been supported by the Air Force Office of Scientific Research, the Office of Naval Research, the National Science Foundation, and IBM.

References

Barbella, D. and Forbus, K. (2011). Analogical Dialogue Acts: Supporting Learning by Reading Analogies in Instructional Texts. Proceedings of the Twenty-Fifth AAAI Conference on Artificial Intelligence, 1429-1435, San Francisco, CA.

Barbella, D. and Forbus, K. (2013). Analogical Word Sense Disambiguation. Advances in Cognitive Systems, 2:297–315.

Barbella, D. & Forbus, K. (2016) Exploiting Connectivity for Case Construction in Learning by Reading. Advances in Cognitive Systems 4:169–186.

Besold, T., Kuhnberger, K., Plaza, E. (2015) Analogy, Amalgams, and Concept Blending. *3rd Annual Conference on Advances in Cognitive Systems*.

Blass, J. and Forbus, K.D. (2015). Moral Decision-Making by Analogy: Generalizations vs. Exemplars. Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence, 501-507, Austin, Texas.

Blass, J. A. & Forbus, K. D. (2016) Modeling Commonsense Reasoning via Analogical Chaining: A Preliminary Report. Proceedings of the 38th Annual Meeting of the Cognitive Science Society, 556-561, Philadelphia, PA, August.

Catrambone, R. (2002). The effects of surface and structural features matches on the access of story analogies. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, 28, 318–334.

Chang, M.D. (2016). Capturing Qualitative Science Knowledge with Multimodal Instructional Analogies. Doctoral dissertation, Northwestern University, Department of Electrical Engineering and Computer Science, Evanston, Illinois.

Chang, M. D. and Forbus, K.D. (2012). Using Quantitative Information to Improve Analogical Matching Between Sketches. *Proceedings of the 24th Annual Conference on Innovative Applications of Artificial Intelligence (IAAI)*, 2269-2274. Toronto, Canada.

Chen, K. and Forbus, K.D. (2018). Action Recognition from Skeleton Data via Analogical Generalization over Qualitative Representations, 638-645. Proceedings of AAAI 2018.

Dannenhauer, D., & Monoz-Avila, H. (2013). LUIGi: A Goal-Driven Autonomy Agent Reasoning with Ontologies. *Proceedings of the 2nd Annual Conference on Advances in Cognitive Systems*.

Domingos, P. (2015) *The Master Algorithm: How the Quest for the Ultimate Learning Machine will Remake Our World*. New York: Basic Books.

Fillmore, C., Baker, C. & Hiroaki, S. (2002). The FrameNet Database and Software Tools. *Proceedings of the 3rd Int. Conf. on Language Resources and Evaluation, Vol IV,* 371-375. Las Palmas. LREC.

Fitzgerald, T., Bullard, K., Thomaz, A., & Goel, A. (2016). Situated Mapping for Transfer Learning. *Advances in Cognitive Systems 4.*

Forbus, K. (1984). Qualitative process theory. *Artificial Intelligence*, 24, 85–168.

Forbus, K. D., Chang, M., McLure, M. & Usher, M. (2017a). The Cognitive Science of Sketch Worksheets. *Topics in Cognitive Science*. DOI: 10.1111/tops.12262

Forbus, K., Ferguson, R., Lovett, A., & Gentner, D. (2017b). Extending SME to Handle Large-Scale Cognitive Modeling. *Cognitive Science*, 41, 1152-1201. DOI:10.1111/cogs.12377.

Forbus, K.D., Garnier, B., Tikoff, B., Marko, W., Usher, M. & McLure, M. (2018). Sketch Worksheets in STEM Classrooms: Two Deployments. Deployed Application Prize paper. *Proceedings of the 30th AAAI Conference on Innovative Applications of Artificial Intelligence*, 7665-7672, New Orleans.

Forbus, K., Gentner, D. and Law, K. 1995. MAC/FAC: A model of Similarity-based Retrieval. *Cognitive Science*, 19(2), April-June, pp 141–205.

Forbus, K. & Hinrichs, T. (2018) Analogy and Relational Representations in the Companion Cognitive Architecture. *AI Magazine*, 38(4), pp. 34-42.

Forbus, K., Klenk, M., and Hinrichs, T. 2009. Companion Cognitive Systems: Design Goals and Lessons Learned So Far. *IEEE Intelligent Systems*, 24(4), 36–46.

Forbus, K., Liang, C., & Rabkina, I. (2017c) Representation and Computation in Cognitive Models. *Topics in Cognitive Science*. 9(3):694-718

Forbus, K., Mostek, T. and Ferguson, R. (2002). An analogy ontology for integrating analogical processing and first-principles reasoning. *Proceedings of the 14th AAAI Conference on Innovative Applications of Artificial Intelligence*, 878-885.

Gentner, D. (1983). Structure-mapping: A theoretical framework for analogy. *Cognitive Science*, 7, 155–170.

Guha, R.V.: Contexts: a formalization and some applications. Technical Report STAN-CS-91-1399, Stanford CS Dept., Stanford, CA (1991)

Gust, H., Kuhnberger, K., & Schmid, U. (2004) Ontological Aspects of Computing Analogies. *Proceedings of the 6th International Conference on Cognitive Modeling,* 350-351

Hinrichs, T. and Forbus, K. (2007). Analogical Learning in a Turn-Based Strategy Game. *Proceedings of the Twentieth International Joint Conference on Artificial Intelligence*, 853-858. Hyderabad, India.

Hinrichs, T. and Forbus, K. (2011). Transfer Learning Through Analogy in Games. *AI Magazine*, 32(1), 72–83.

Hinrichs, T., and Forbus, K. (2015). Qualitative models for strategic planning. *Proceedings of the 3rd Annual Conference on Advances in Cognitive Systems.*

Holmes, D. & Winston, P. (2016). Story-enabled hypothetical reasoning. *Advances in Cognitive Systems 4*.

Holmes, D., & Winston, P. (2017) Character-building stories. *Advances in Cognitive Systems 5.*

Hummel, J. E., Licato, J., & Bringsjord, S. (2014). Analogy, explanation, and proof. *Frontiers in Human Neuroscience*, 8:867.

Kandaswamy, S., Forbus, K., and Gentner, D. (2014). Modeling Learning via Progressive Alignment using Interim Generalizations. *Proceedings of the Cognitive Science Society*, 2471-2476.

Klenk, M. and Forbus, K. (2009). Analogical Model Formulation for AP Physics Problems. *Artificial Intelligence*, 173(18), 1615–1638. doi:10.1016/j.artint.2009.09.003

Laird, J. (2012) *The SOAR Cognitive Architecture*. MIT Press.

Lockwood, K. and Forbus, K. (2009). Multimodal knowledge capture from text and diagrams. *Proceedings of the 5th International Conference on Knowledge Capture (KCAP-2009)*, 65-72. Redondo Beach, CA.

Lockwood, K., Lovett, A., and Forbus, K. (2008). Automatic Classification of Containment and Support Spatial Relations in English and Dutch. In Freksa, C., Newcombe, N., Gardenfors, P. Wolfl, S. (Eds.) *Spatial Cognition VI: Learning, Reasoning, and Talking about Space. (Spatial Cognition 2008).* Lecture Notes in Computer Science, Volume 5248, 283-294. Springer, Berlin.

Lovett, A., & Forbus, K. (2017) Modeling visual problem solving as analogical reasoning. *Psychological Review*, 124(1), pp. 60-90.

McFate, C., and Forbus, K. (2016). Analogical Generalization and Retrieval for Denominal Verb Interpretation. *Proceedings of the 38th Annual Meeting of the Cognitive Science Society*, 1277-1282, Philadelphia, PA, August.

McLure, M. and Forbus, K. (2012). Encoding Strategies for Learning Geographical Concepts via Analogy. *Proceedings of the 26th International Workshop on Qualitative Reasoning*. Los Angeles, CA.

McLure, M.D., Friedman S.E. and Forbus, K.D. (2015). Extending Analogical Generalization with Near-Misses. *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, 565-571, Austin, Texas.

McShane, M., Nirenburg, S., Beale, S. and Johnson, B. 2012. Resolving elided scopes of modality in OntoAgent. *Advances in Cognitive Systems,* 2: 95-112.

Mostek, T., Forbus, K, and Meverden, C. (2000). Dynamic case creation and expansion for analogical reasoning. *Proceedings of AAAI-2000*, 323-329. Austin, TX.

Pierce, C., Booth, D., Ogbuji, C., Deaton, C., Blackstone, E., & Lenat, D. (2012). SemanticDB: A Semantic Web infrastructure for Clinical Research and Quality Reporting. *Current Bioinformatics*, 7(3), DOI : 10.2174/157489312802460730

Rabkina, I., McFate, C., Forbus, K. D., & Hoyos, C. (2017). Towards a Computational Analogical Theory of Mind. In *Proceedings of the 39th Annual Conference of the Cognitive Science Society*, 2949-2954, London, England.

Rabkina, I., McFate, C., & Forbus K. D. (2018). Bootstrapping from Language in the Analogical Theory of Mind Model. In *Proceedings of the 40th Annual Conference of the Cognitive Science Society,* Madison, Wisconsin.

Riesbeck, C. & Schank, R. (1989). *Inside Case-Based Reasoning*. Psychology Press.

Vattam, S., & Goel, A. (2012). Interactive Analogical Retrieval. *1st Annual Conference on Advances in Cognitive Systems*.

Veal, T. (2006) Re-representation and creative analogy: A lexico-semantic perspective. *New Generation Computing*, 24(3):223-240.

Weller, S. & Schmid, U. (2006). Analogy by Abstraction. *Proceedings of the 7th International Conference on Cognitive Modeling*, 334-339, Trieste.

Winston, P. H., 1970. Learning structural descriptions from examples. Ph.D. diss., MIT, Cambridge, MA

Wolff, P., & Gentner, D. (2011). Structure-mapping in metaphor comprehension. *Cognitive Science*, 35. 1456-1488.