Making Clouds from Cement: Building Abstractions around Concrete Examples

Kenneth B. Haase
MIT Media Laboratory

This article discusses a radically memory-based representation where matches between concrete examples — rather than translation to canonical form and subsumption under abstract definitions — serves as the basic element of a representation. In this system, representation is a problem solving process of indexing and matching descriptions; this process uses the results of past representational problem solving to support current matching and indexing. New representations emerge as new examples fail to find clear analogs among existing descriptions. This analogical representation is currently being applied to story-understanding and video representation in projects at the Media Laboratory.

1 Overview

This article presents an alternative to the traditional equation of 'understanding' with translation to a uniform canonical representation. In an *analogical representation*, understanding is a process of drawing analogies — usually mundane ones — between new descriptions and existing descriptions. The 'meaning' of terms in an analogical representation comes from analogies among existing descriptions rather than from formal symbols with some attributed semantics. In an analogical representation, phenomena such as abstraction and categorization arise from the processes of indexing and analogizing among concrete examples rather than being codified into internal structures.

The motivations for an analogical account of representation include:

- avoiding the need to posit special internal mechanisms and structures supporting abstract schemata
- allowing the construction of partially competent systems without the intervention of a representation designer
- providing a locus for interpretive flexibility when dealing with incomplete, inconsistent, and incommensurable representations

This final motivation is especially important at the crossroads of vision and language where radically different underlying representations come together. Linguistic and visual languages emphasize different attributes, focus on different relations, and in general resist translation to a canonical form.

However, the problems involved in replacing canonical representations with memory based representations aren't trivial either, since much of the power of knowledge based systems comes from their canonical representations and explicit abstractions. In particular,

- canonical representations enable the simple specification of procedures with rich behavior over a wide range of circumstances
- transfer of expertise typically relies upon the indexing of distinct situations to identical generalizations
- storage considerations plague systems depending on a background of cases, since a few abstract definitions may compactly summarize the generalizations emergent from a plethora of individual cases.

In this article, I describe Mnemosyne, an analogical representation system being developed at the Media Lab for a variety of description, indexing, comprehension, and reasoning tasks. I open with an overview of Mnemosyne's design followed by a discussion of its motivations and responses to the most obvious criticisms.

I follow this discussion with a more detailed account of its Mnemosyne's implementation, centered around a simple example of language/meaning accquisition (learning to match passive and active sentences). This was chosen as the hardest compact example for an analogical representation because though the thematic contents of active and passive sentences are largely the same, their surface forms are symmetric opposites. I show how Mnemosyne's mechanism allow the system to accurately interpret the relations between such sentences without any explicit case-frame like abstractions.

Finally, I briefly discuss ongoing and planned work in scaling up Mnemosyne to work with large databases of text, still image, and full-motion video.

2 A Sketch of Mnemosyne

In this section, I sketch Mnemosyne in a few broad strokes. In later sections, I will fill this in with more detail and cross-hatching in a few choice places. This sketch of Mnemosyne contrasts its with more conventional representations of which CYC [Lenat and Guha, 1990] might be a paradigmatic example. In Mnemosyne:

- All descriptions are contextual; the identity of individuals across different contexts must be explicitly stated rather than being implicitly encoded
- Significance comes from the prior description to which a description is indexed
- Analogy is a case of contextual indexing; indexing an element a of X to an element b of Y is saying that a plays a similar role in X as b plays in Y.
- past indexing and matching enables current indexing and matching by both the indexing of distinct descriptions to common prototypes (past examples) and the use of explicit records of difficult to reproduce indexing steps.

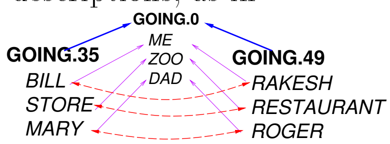
The contextuality of all descriptions means that 'Romeo' in the sentence 'Romeo loved Juliet' is not represented by some unique global 'Romeo' descriptor, but instead represents only the facts about Romeo relevant to the context of his love for Juliet. The unity of the various Romeo descriptions comes from regularities in the indexing structure and not from their representation in a single symbol. This is important because the role played by Romeo — and the analogies which may be drawn from him — is a function of his context.

By separating out the facts about Romeo to the contexts in which they reside, the possibility is left open of redividing the understanding of Romeo depending on circumstance or convinience. One can see Romeo as a tortured lover, as a fumbling adolescent, as a handsome actor, or as a symbolic figure. More pragmatically, one can see a chair as an object to be sat upon, an obstacle to be avoided, or as a fixture to be moved from room to room.

Experience usually gives us our experiences conveniently contextualized and translations to canonical representations often throw this information away. By keeping it, the problem of figuring out which elements of a canonical description are relevant to a particular context is obviated; all descriptions are inherently contextual and the problem is instead one of integrating the descriptions from their different contexts. This is similar to Agre and Chapman's arguments in favor of indexical representations [Agre and Chapman, 1987, Agre, 1988, Chapman, 1991]. It also bears some similarity to Minsky's notion of c-lines [Minsky, 1977] or **pronomes** where aspects of siutations are placed topologically near the mechanisms which use them. I don't pretend that the problem of determining identity in such a fragmented representation is easy, but I believe it easier than teasing apart the very different descriptions assigned to some unique individual.

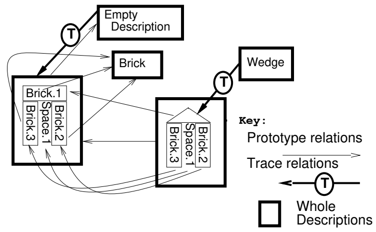
'Meaning' in Mnemosyne is based on the indexing of these contextualized descriptions to other contextualized descriptions. The description to which a description is indexed is called its 'prototype'[1] This characterization of meaning as indexing extends to a treatment of analogy as 'contextual indexing'; a description is not merely indexed to descriptions like it in some decontextualized sense, but to descriptions which play roles similar (in their contexts) to the role it plays in its context.

The process of figuring out new analogies is based on past analogies. This occurs in two ways: through the identification of common prototypes and through records of past analogical problem solving stored in the trace network. Common prototypes support translation between descriptions, as in



where the analogies drawn from GOING.49 and GOING.35 to GOING.0 enable analogies between GOING.49 and GOING.35. This uses the results of past indexing and analogizing to support further analogies. This indexing process uses a 'trace memory' to store the significant variations of a particular prototype. A description may be a variation because it differs structurally from its prototype or because some matching process wrenched its from its prototype to another distinct prototype. Here, the left hand trace (on the empty description) records a structural variation, while the right hand trace (on Wedge) records a radical shift of prototype:

[1]The connotations of the term 'prototype' are unfortunate, but I have not yet come upon a better one. Distinct from its more conventional use, prototypes in Mnemosyne are not distinguished from other instances; however the most obvious alternative, 'exemplar,' fails to capture the way in which Mnemosyne's prototypes determine the structure of other descriptions. Ideally, I would like a term which is more general than 'prototype' but narrower than 'exemplar'.



Used in this way, the trace network combines the functions of a discrimination network and a reminding network [Bareiss, 1989]; the presence of a trace (such as that between some prior wedge and 'wedge as lintel') biases future analogizing.

By these mechanisms, Mnemosyne moves a variety of deep representational functions — canonicalization, abstraction, identification — out of the implicit structure of the representation and into the processes of perception and comprehension. The following sections describe the value of such a move and address some of the more obvious arguments against it.

3 Motivation

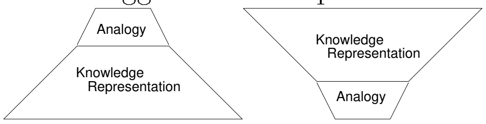
As sketched above, the motivations for an analogical account of representation include:

- avoiding the need for special internal mechanisms and structures supporting abstract schemata
- allowing the construction of partially competent systems without the intervention of a representation designer
- providing a locus for interpretive flexibility when dealing with incomplete, inconsistent, and incommensurable representations

Tis section expands on each of these motivations in turn; after this, the next section likewise expands on the criticisms I mentioned above.

Theoretical Parsimony

From the standpoint of theoretical parsimony, one clear attraction of analogical representations is that they do not require special mental constructions and mechanisms for describing and generating abstractions and canonicalizations. A counter-argument to this might say that we've traded these elegant and uniform mechanisms for the overhead of a particular matching architecture and a huge library of cases. But the ability to represent and compare instances is a known capacity of intelligence, which is usually modelled as built on top of canonical and abstract representations. If I can show that by turning this pyramid upside-down, I can build everything upon a smaller base, Occam's choice suggests the simpler basis.



I addition, there is developmental support for a model which specifies formal abstractive behavior as emergent from simpler abilities for dealing with concrete instances. Piaget's crucial distinction between concrete and formal operations is based on the observation that younger children's apparently fluid abstractions dissolve when pushed beyond the concrete arena. If one assumes that a formal representation — with its symmetries and equivalence relations — exists as the basis for concrete activity, one needs to explain why its form is not consistently revealed in childen's actions and deliberations. On the other hand, if one considers such formal and canonical representation as emergent from more basic abilities for combining concrete examples, this phenomenon is not only unsuprising, it is predicted!

To back off a little from the most radical empirical stance, I think that substantial initial structure is probably imposed on descriptive encodings by innate perceptual and cognitive structures. However, the human ability to easily appropriate and adapt representations to new purposes suggests some fundamental mechanism for building new representations around existing representations whether innate or derived.

Knowledge Acquisition

But theoretical parsimony aside, analogical representations also have purely engineering advantages. The most obvious of these is that the problem of translating experience into 'knowledge' disappears; its place is taken by a simple (but not arbitrary) translation of 'experience' into some segmented set of descriptions close to the original data. But keeping some priniciple of 'intellectual conservation' in mind, the obvious question is 'where has all the knowledge gone?'

The knowledge has migrated into two places: the indexing structure and the matching mechanism. In Mnemosyne in particular, this means the protoypical relations between descriptions and sub-descriptions and the variations (and records of past analogizing) stored in the trace network.

This knowledge is built up by the program's own matching efforts and by the tutoring (correction and hints) of a human overseer. An important point is that this knowledge is deployed in the context of a variety of matching heuristics; this means that in tutoring the system one need not enumerate the space of translations exhaustively, but need only 'nail down' enough places that the system's own (usually more conservative) matching heuristics can do the rest. This gives descriptions in an analogical representation a curious character; while discrete and clearly symbolic, they nonetheless exhibit much of the malleability of more 'analog' connectionist representations.

One advantage of knowledge-based systems touted by their early proponents was the simplicity of the matching and rule-firing mechanism relative to more baroque schemes. However, this design decision merely shifted the complexity of matching into the human process of representation design and knowledge accquisition. Of course it worked better, as it was running on a smarter processor (often a graduate student), but in the long run — I argue — a vital autonomy was lost.

Interpretive Flexibility

The lost autonomy lay in the ability to use representations *interpretively* and a consequent loss of the ability to deal with multiple conflicting representations. It is a hallmark of human flexibility that when stuck, a human problem solver or learner can consider the possibility that its representation is flawed in some way and either correct it or blend it with some other locally more suitable representation.

A memory based representation has a certain interpretive flexibility because the description of a property (a descriptive element in a prticular context) is distinguished from the description of its significance (the family of other descriptions to which it is analogous). The determination of the significance of a description then becomes a process subject to background knowledge, past cases, and the exegiences of the current circumstance. In this process, certain elements can be left temporarily or permanently unresolved, roles may be swapped, or matches allowed which are structurally unfamiliar but functionally clear (for instance, the single wheeled based replacing four individual sticks supporting a chair.)

Routine cognition relies on interpretive flexibility; [Winograd and Flores, 1986] describe example after example where representations are used interpretively and argue from a rich philosphical and linguistic tradition that all representations are fundamentally interpretive. Some (especially [Dreyfus, 1979] and to a lesser extent [Winograd and Flores, 1986]) argue that this characteristic of human activity is a persuasive argument against symbolic approaches to general intelligence. Analogical representation presents a third alternative which is unabashedly symbolic (there are discrete hierarchical descriptions) but nonetheless provides an interpretive account of representation.

4 Criticisms

This section address a few of the most obvious criticisms of analogical representations. These responses build on one another so that earlier ones are substantially longer than the later ones. Other criticisms certainly could be (and certainaly will be) made, but I have selected here is on the most obvious ones.

Canonical representations enable the simple specification of procedures with rich behavior over a wide range of circumstances

The notion of an intelligent program without a canonical internal representation seems at first as ludicrous as the idea of a car without wheels. Canonical representations are what enables astonishingly simple rules to exhibit rich behavior and to operate over a broad range of situations. In designing intelligent systems, a major element of the process is the specification of a representation where salient knowledge is made explicit and over which simple syntactic processes can produce effective results. To argue for a model of activity where such representations are excluded takes from the system designer one of her most powerful levers for the aplification of design effort.

As I address these criticisms, I may seem to be speaking out of two sides of my mouth as I argue that: (a) one can see a form of a canonical representation lurking in memory based representations but (b) canonical representations are inherently flawed as a basis for intelligent behavior. The reason for this contradictory juxtaposition is the argument that analogical representations ae like canonical representations in the important ways and also *unlike* canonical representations in equally important ways.

An analogical representation can be considered as a canonical representation where the 'canon' consists of prior descriptions and their elements and translation to the canon is the indexing of new descriptions to these prior descriptions and elements. This indexing of elements distinguishes analogical representations from exemplar-based accounts of categorization.[Bareiss, 1989] Exemplar-based categorization typically assumes a base feature language for descriptions and builds categories around exemplary instances described in this feature language. In an analogical representation, the elements of descriptions also get their significance by analogy to other descriptive elements; i.e. it's analogy and examples 'all the way down.'

In a conventional canonical representation, categories and roles are described by tags ascribed to descriptions and their elements. The problem with such tags are:

- they require positing a specialized internal tag language
- they require specialized mechanisms for figuring out what the tags are
- they demand a distinct learning process for extending the tag space as examples and variations arrive from the domain
- the tags by themselves are 'decontextualized' by their separation from the space of descriptions

These are avoided in the analogical flavor of canonical representation:

- tags are simply pointers to previous descriptions and descriptive elements

- the assignment of tags is simply the indexing of descriptions and their elements in the space of descriptions
- learning is integrated with perception and comprehension and the failure of such processes extends the representation; when an example fails to fit the 'canon,' the 'canon' is automatically extended by the failure
- categories are inherently contextual; this model of category formation contrasts with conventional category formation in much the same way that situated action contrasts with the plan-and-execute model of activity.

When descriptions have been indexed, the cost of determining equivalence is only the relatively small cost of marker-passing through the prototype hierarchy (linear in its depth) which is comparable to the constant time comparsion of literal tags. Thus, procedures which operate over an analogical representation may work just as effectively as if operating over a canonical representation.

Transfer of expertise typically relies upon the indexing ot distinct situations to identical generalizations

As mentioned above, historically prior descriptions in an analogical representation play the role of the explicit canonicalizations or generalizations in a conventional representation language. Thus, there is still a 'common ground' for the transfer of expertise. The deeper question behind this is the ability to have descriptions which match a whole space of other descriptions.

Because generalizations are not explicitly represented in an analogical representation, generalization is shifted into the processes of matching and indexing. Generalizations emerge dynamically in an analogical representation as the system attempts to deal with new descriptions. The basis for such generalizations — the knowledge underlying them — lies in the traces of past matching and indexing activity. In Mnemosyne, these traces takes two forms: the prototypes which indexing and matching eventually determine for individual descriptions and their elements; and records of past matching activity which memoizes those matches taking substantial effort (or insight or advice) to produce.

In the example below, I will show in more detail how abstraction emerges from the indexing and matching processes and how the results of past indexing and matching are reused in new contexts.

Storage considerations plague analogical systems

This final criticism of analogical representation is undeniable but largely inconsequential. an analogical representation does indeed have substantially greater space demands —- for the storage of cases — than conventional approaches. While this is true, in many cases those redundant cases need to be stored separately because they are not really completely identical and future learning may need to look back on what passed unnoticed in the past. In addition, the argument about space has relatively little relevance when considering human mental capacity. And finally, a degree of 'case compression' can be achieved by removing information which could be regenerated based on the descriptions to which a given description is indexed. Such a scheme would reduce each example to only the differences between it and te nearest previous example.

5 Mnemosyne: Underlying Structures

This section describes the structures underlying Mnemosyne and its basic mechanisms. The subsequent section offers a detailed example which shows these various structures and mechanisms in action.

Mnemosyne is implemented on top of FRAMER[Haase, 1993], a persistent object and inference system developed at the Media Lab for a diverse range of representation projects. For purposes of this article, I will simply sketch the aspects of the structure used by Mnemosyne.

The basic descriptive element in FRAMER is the frame; a frame has a set of named **annotations** (each themselves a frame), a single **prototype**, and a **ground** which points to either another frame, a LISP object, or a set of such frames or objects. In general, the annotation structure provides a hierarchical organization of frames (essentially, a partonomy [Langley and Thompson, 1991]), the prototype relation specifies an inheritance relations between elements and subsets of this hierarchy, and the ground relations specify connections within the hierarchy. Figure 1 shows an example of FRAMER structure being used to support a conventional knowledge representation language.

FRAMER itself provides a simple inheritance mechanism (the 'default prototype rule') based on naming of annotations: if Y is the prototype of X, the prototype of the annotation named 'a' of X is the annotation named 'a' of Y. Mnemosyne encodes analogical correspondence by the prototype relation; if the element p of X is analogous to the element q of Y, then the prototype of p is q. Given this characterization, the default prototype rule implements the 'nominal correspondence' matching heuristic implicit in most knowledge representation languages: concepts with the same name in the different contexts are analogous.

How Mnemosyne uses FRAMER

Mnemosyne matches FRAMER frames, using the annotation structure to implement descriptive components, the prototype relation to indicate analogous elements, and the ground relation to describe relations between components. In the examples

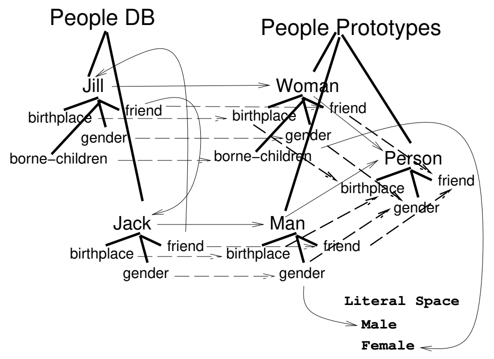


Figure 1: FRAMER has three distinct organizational axes: annotation relations between frames (bold lines up and down), prototype relations between frames (slender arrows sideways), and groundings connecting frames to constant values or to other frames (bold arrows). The dashed prototype relations indicate those relations which obey the default prototype rule.

below, it is important to realize that the components of frames are entirely contextual; 'Romeo' in the sentence 'Romeo loved Juliet' is not represented by some unique global 'Romeo' descriptor, but instead represents only the facts about Romeo relevant to the context of his love for Juliet. The unity of the various Romeo descriptions comes from regularities in the indexing structure and not from their representation in a single symbol.

FRAMER's prototype link is used to encode analogical references and to define the **cognate** relation which is the basis for Mnemosyne's matching mechanisms. Two descriptive elements are cognates if there exists some prototype common to them which is common to no other pairings of other elements from the descriptions. Note that this relation is contextual and non-monotonic: elements within a context 'compete' for cognates and the addition of a new element may invalidate existing cognate relations. Assuming some constant bounding the depth of the prototype hierarchy, cognates can be determined in time linear in the size of the descriptions compared; in fact, all the cognate relations between two descriptions can be computed within the same $O(n)$ bounds, allowing an amortized constant time algorithm for determining cognates.

Matching, Indexing, and Adapting

Mnemosyne consists of three major components: a matcher which compares descriptions, an indexer which given one description tries to find a god existing description with which to match it, and an adatper which given two descriptions which (possibly partially) match, transfers structure and constraint from one to the other.

Mnemosyne uses a hierarchy of procedural matchers to compare frames; the ordering in the hierarchy is determined by the complexity of the corresponding matchers and cheaper matchers are tried first. Each matcher assumes the matchers below it and the bottom level of matching is based on the cognate relation described above: elements are matched to their cognates. When cognate matching fails, Mnemosyne invokes a number of more expensive mechanisms in order of increasing complexity:

- **structure matching** matches elements based on the recursive matching of cognates;
- **trace matching** matches elements based on previous match failures and their resolutions
- **hypothetical matching** matches elements based on structural systematicity (after [Gentner, 1983]).
- **tutored matching** applies a range of other mechanisms, including (but not limited to) interaction with a human tutor.

The indexing process is built on the matching process and proceeds bottom up; a new description is indexed by first indexing its components, as in Labyrinth [Langley and Thompson, 1991]. This results in a set of possible prototypes for the elements; the contexts in which these possible prototypes occur provides the seed for a search for prototypes for the elements embedding context (the description itself). This search follows the trace network (which records partially failed matches and so may point to better matches). Where the component matching process starts depends on the application domain; in the language example below and in my group's current research on example-based retrieval, I assume that words start being indexed from the instance of their first occurence in any text or from some ur-'Word' otherwise.

The adaptation process operates on the results of indexing (and thus of matching) in a number of ways:

- it changes the prototype links of the description and its sub-descriptions to point to the descriptions to which they are indexed;
- it prunes the grounds of the new structure to remove any ambiguities clarified by the new prototypes;
- it leaves a 'trace' whenever a change of prototype is not one that would naturally result from structure based indexing.

As in any dynamic memory, storing an instance in memory changes both the structure of the instance and the structure of the memory. The changes in the memory are of two sorts: the increasing variety of prototypes built up by new examples and the increasing number of traces to support the indexing process.

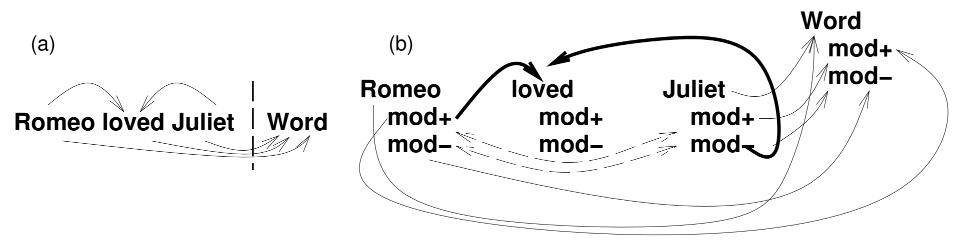


Figure 2: A modification graph representation of the sentence 'Romeo loved Juliet' and its FRAMER implementation.

6 An Example: Language Learning

In this section, I detail an instance of example based language accquisition in Mnemosyne. This domain was chosen for its familiarity and because it presents some of the most obvious challenges for an example based approach: 'meaning' is sometimes quite different from 'form'. The climax of the example will be the accquistion of the ability to draw appropriate analogies between passive and active sentences even though their syntactic structures are actually inverses of one another. What I am *not* arguing in this example is that language learning can be completely explained as the empirical accumulation of examples in an indexed memory. (Though I would be willing to argue that some of it is.) Most of the example is simply a demonstration that abilities which were previously assumed to require canonical abstractions can be done entirely with concrete examples.

The representation used for sentences are 'modification graphs' linking descriptions of the words in a sentence; modification graphs are much like the dependency graphs described in [Meĺčuk, 1988]. Figure 2(a) shows a modification graph representation of the sentence 'Romeo loved Juliet'. The arcs above the sentence indicate the modification graph relationships used to describe sentence structure; the arcs below the sentence indicate prototype relations between word representations. Words are linked to other words by pointers in their 'mod+' and 'mod-' annotations; a word's mod+ pointers indicate the succeeding word(s) which it modifies while its mod- pointers indicate the preceding word(s) which it modifies. When the ground of a frame (annotation) points to more than one other frame, this is intepreted as an ambiguity; when Mnemosyne attempts to structurally match such ambiguous descriptions, if any choice from the ambiguious combination will allow a match, the match is made.

Figure 2(b) indicates the FRAMER structure underlying what appears in (a): 'Romeo,' 'loved,' and 'Juliet' are all elements of the description 'Sentence.1'; their prototype links point to some prototype frame 'Word'; and the ground of their annotations (the bold arcs) indicate links between words. These annotations have their own pototype links to annotations of 'Word' and these prototype links determine

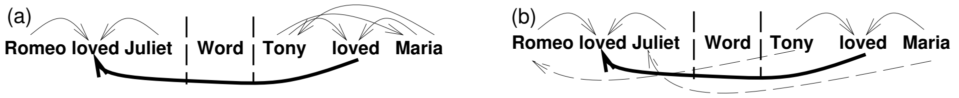


Figure 3: Mnemosyne learns to interpret active sentences by a structural match between subjects and objects.

a cognate relation between the various 'mod+' and 'mod-' annotations; the dashed lines show two of the cognate relations thus determined. Due to these cognate relations, matching two words matches their modification relations (in their respecive sentences) in both directions.

Understanding a simple sentence

When a new sentence is encountered (Figure 3), Mnemosyne attempts to match it to past sentences with similar words; [2] the bold arc in the figure is the prototype relation connecting the new occurence of 'loved' with the old. It is also the only cognate relation between the two sentences since all the other relations pass ambigously through the 'Word' prototype (which is ommitted for clarity). In addition, the modification graph for the new sentence (Figure 3(a)) ambiguously represents all possible connections. Since cognate matching is insufficient, structural matching steps in and finds a match between the subjects and objects (the dotted lines in Figure 3(b)); a structural match is possible since both Maria and Juliet 'might' modify the preceding word 'loves' and both Tony and Romeo 'might' modify the succeeding word 'oves' (but not the other way around). Once determined, the match is established by making a new set of prototype relations and pruning the annotations of Tony and Maria to be consistent with their new prototypes. This disambiguates the original sentence and establishes its 'meaning' in two ways: the pruning corrects the modification graph and the change of prototypes distinguishes the role played by each element of the sentence. **Thus, matching to the background can provide structure to structure new instances based on the structure of previous cases.**

Learning the Passive of one verb

One compelling argument for canonical form is that syntactically distinct forms often have identical meaning; in this section I show how such similarity judgements can be made without recourse to canonical form. In Figure 4, I present the system with the passive form of an assertion which it had previously seen in the active sense. The similarity of words establishes prototype relations which determine cognate relations capturing the sentence 'meaning' (Figure 4(a)); when a new passive sentence is

[2]This is a simple case of the bottom up indexing; where the system has built up a trace network indicating functionally equivalent words, this includes such sentences

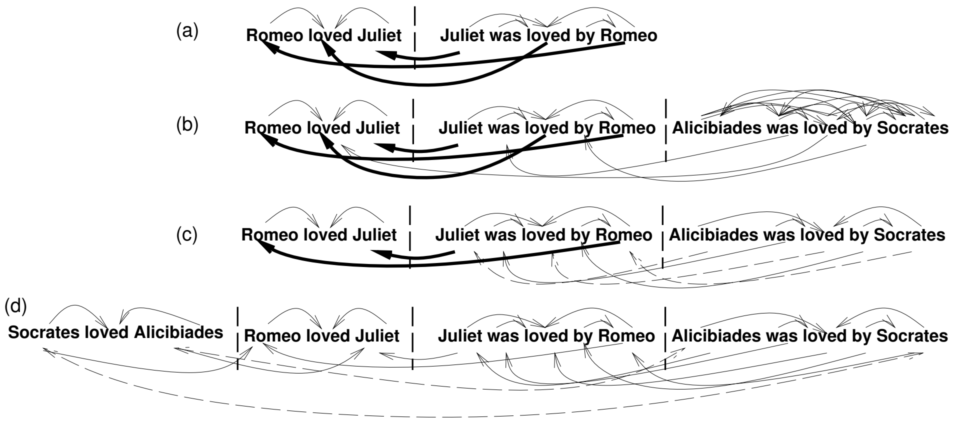


Figure 4: Mnemosyne learns to interpret passive sentences by matching a passive version of a sentence to the meaning-equivalent active version it has seen before.

encountered (Figure 4(b)), it is indexed to the previous one and a match proceeds just as in Figure 3. This results in a disambiguated sentence with prototypes in the original passive example (Figure 4(c)). But the prototype relations drawn to this example also enable a cognate relation between this passive sentence and a meaning-identical active one.

The reader may be wondering at this point whether an example like ‘Juliet loved Romeo’ would break the system by teaching it an equivalence between subjects and objects in general. The answer is no and the reason is that what is learned by the formation of prototype rlations (analogies) is limited to the descriptions whose prototypes are changed and the description of ‘Juliet’ in each context is unique; shifting its prototype does not effect other occurences of Juliet. The shift of prototype would assert (more or less) ‘This Juliet isn’t like the Juliets I’ve seen before.’ **The move from a unitary representation for individuals (as in most representation languages) to a contextual representation allows us to learn rules for equivalence *contextually* and to only refer to such examples when their context merits it.**

Introducing a Different Verb

The example above works for the verb ‘loved’ but what about other verbs? The answer is that when a match cannot be determined, the matcher tries harder and implicitly generalizes the case it has been given. To illustrate this, I first give the system an example of a sentence with a different verb (and meaning) than those I have given it before. I then show how this example and the example of the passive (implicitly generalized) are combined to produce the correct interpretation of the passive.

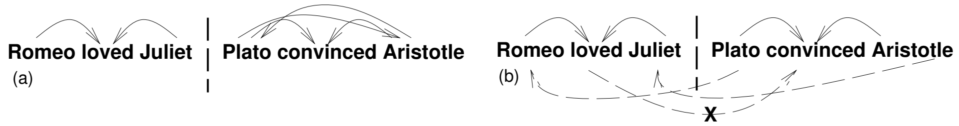


Figure 5: Mnemosyne's 'ontology' is built up by the correction of incorrect analogies, leading to the snapping of links (such as the one here between 'convinced' and 'loved') made incorrectly.

I introduce the verb 'convinced' with the sentence 'Plato convinced Aristotle' in Figure 5; the new sentence has no obvious cognates with the initial sentence (but was indexed there bcause there were no better alternatives); since there are no cognates, there can be no structural matches. Since I do not have a background of past matches in the trace memory, I invoke the hypothetical matcher which tells us that if 'convinced' were analgous to 'loved' we'd have two fine structural matches between subject and oject. Unfortunately, this is an incorrect analogy and some outside source intervenes and snaps the erroneous link, giving us the situation in Figure 5(b).

This correction could come from either a human tutor or some higher level matcher trying to reconcile other relations in the story. As a result of the link being snapped, the occurence of 'convinced' becomes a description without a prototype. **Mnemosyne's 'base ontology' — such as it is — consists of just such prototype-free descriptions created whenever a mistaken analogy is corrected and a new distinction introduced.**

Generalizing the Passive

Given this different verb and its representation, how can the grammatical knowledge about the passive be transferred to it? If we introduce the sentence 'Kant was convinced by Rousseau, its elements have prototypes in both the initial introduction of 'convinced' and the initial introduction of the passive. (Figure 6(a)) Neither is obviously a better candidate based on prototype relations alone, but a hypothetical match can again b found between 'convinced' and 'loved' which allows a match of subject and object (Figure 6(b)), while there is no such 'crystallizing hypothesis' with the original occurrence of 'convinced'. However, once this match to the passive prototype is accomplished, there is a cognate relation (indicated by bold lines) to the left-hand sentence through the original 'loved' sentence (Figure 6(c)) which allows us to shift back to it to get a correct interpretation. The motivation for shifting back is a conservative one: matches are preferred which end up with as few changes of prototype as possible. **Once the description's prototype is shifted back (Figure 6(d)), a complete and correct match to the earlier occurence of 'convinced' exists, combining the grammatical knowledge implicit in a pas-**

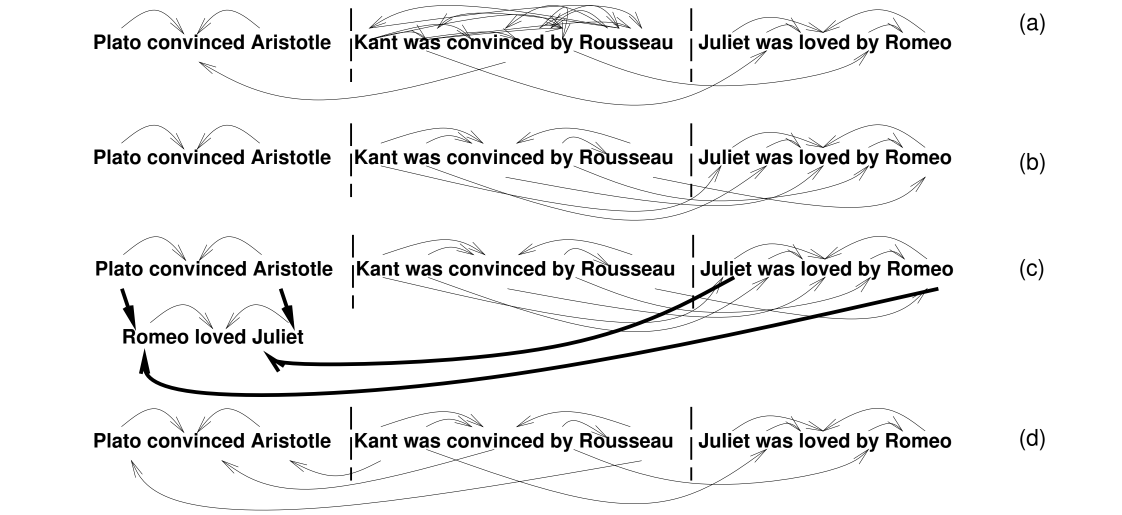


Figure 6: The passive can be generalized by first matching to the grammatical pecedent and then to the semantic precedent; the first match creates the cognate relations which enable the second match, but the second match is preferred because it requires fewer traces be left in memory.

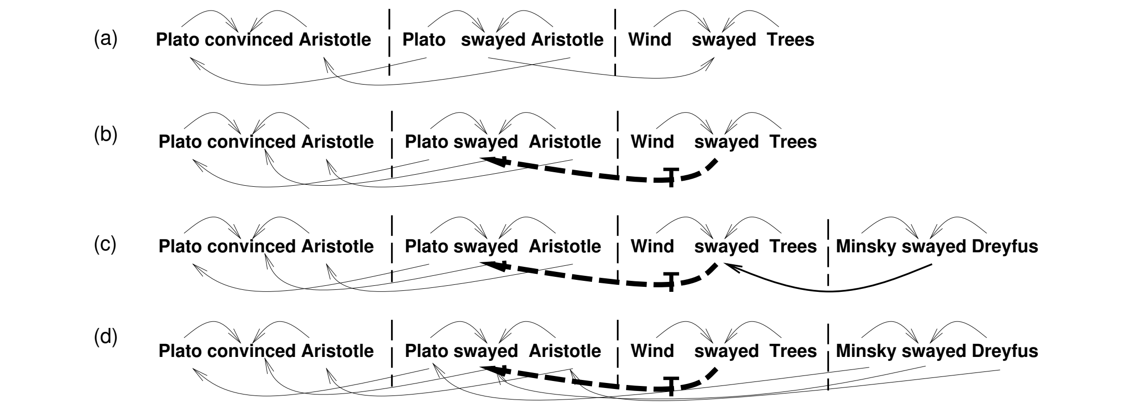


Figure 7: Traces record past analogical reasoning. In this example, the analogical work of figuring out a metaphorical use of the word 'swayed' is saved as a short cut for future uses of the metaphor.

sive sentence with a different verb together with the semantic distinction implicit in the earlier active sense of the same verb.

Learning Synonyms (Contextually)

In the examples above, hypothetical matching was invoked when there were not the cognate matches to enable structural matching; hypothetical matching is potentially quite expensive since it searches an exponential space of pairings and then invokes structure matching as an evaluation function. For this reason, Mnemosyne saves the changes introduced by hypothetical matching (or other mechanisms above hypothetical matching) in a trace memory. To demonstrate how this works, a metaphorical sense of 'swayed' is introduced in Figure 7 which is meant to be analogous to 'convinced'. As in learning the passive, the system is given an example where a meaning interpretation is already known (Figure 7(a)); hypothetical matching draws the correct analogies to the earlier example and leaves a trace (Figure 7(b)) from the earlier usage of 'swayed' to the later usage. Subsequently, when a new example is seen (Figure 7(c)), the matching process follows the trace (Figure 7(d)) to get a correct interpretation.

Traces memoize analogical reasoning in the same way that chunks in SOAR [Laird *et al.*, 1986] memoize production firing or k-lines in Society of Mind [Minsky, 1986] preserve patterns of successful problem solving activity. The background of past analogical reasoning enumerates a space of 'reasonable' analogies to draw in the future.

7 Related Work

This work draws on several schools in artificial intelligence and allied disciplines. In particular, its use of examplars as a sole representation is similar to Protos [Bareiss, 1989]; one innovation of Mnemosyne over Protos is the replacement of the feature language with examplars. In the focus on indexing, Mnemosyne draws on the numerous efforts emerging from Schank's notions of dynamic memory [Schank, 1982, Schank and Riesbeck, 1989]; the chief difference here is the demonstration (suggestive but not conclusive) that abstractions can emerge from exemplar indexing without requiring explicit codification of abstractions.

In analogical reasoning, the distinctive focus of this work is the placement of routine activity and analogical reasoning in the same framework, so that analogy steps in when routine activity breaks down and routine activity reflects past analogical problem solving. The heuristic importance of systematicity and the idea of structural matches are drawn from [Winston, 1979, Gentner, 1983, Falkenhainer *et al.*, 1989]. Past work on analogy generally has depended — at some basic level — on a matching of literals to literals; in Mnemosyne, since the basic matching level is cognate identification, and the base level is enriched with the increasing complexity of the space of memory structures.

Insofar as the locations to which descriptions are indexed constitute 'concepts,' Mnemosyne can be seen as a concept formation system. Its approach to indexing hierarchical descriptions (using the indexing of parts to determine the indexing of their wholes) is similar to that in `Labyrinth` [Langley and Thompson, 1991], except with the use of analogical indexing in place of Labyrinth's featural and structural indexing.

Another influence on this research has come from workers in prototype theory [Rosch, 1973] and the critical analysis of 'definition based' categorization by thinkers such as [Lakoff, 1987]. Though prototype effects are not conclusive evidence of exemplar based descriptions, they do suggest that a definitional account of categorization and cognition is insufficient to the phenomena. Lakoff's response is to suggest a webwork of idealized cognitive models (ICMs) which multiply describe the world; Mnemosyne demonstrates that the abstractive behavior of such models may be possible without assuming an internal representation of them.

8 Current Applications

One central and defining applications focus for our analogical representation work is text understanding based on analogies within a lightly parsed corpora rather than translation to or categorization under some formal schemata. This work involves using a 'multi-scale parser' to generate modification graphs for a large corpus of texts (currently, hundreds of one-paragraph news summaries) and to index and analogize within this corpus under the oversight of human tutors. The 'Metric of Truth' for this project is the amount of human intervention required over time; if the human tutoring required does not decrease substantially with time, it will be clear that other organizing structure is neccessary besides the structure of the language and examples themselves.

In addition, to the text based work, two other projects are looking at using a memory-based representation to index video and image data.

In Media Streams [Davis, 1991], Mnemosyne is used to describe the fine content of video sequences in a multi-media archive. The representation used is a hybrid of a canonical representation and an analogical one. The canonical representations comes from human annotation of video material with icons in a pre-established ontology. Figure 8 shows the icons annotating a particular video segment. These provide the kernels off of which the indexing and matching of segments to segments occurs. The advantage of the hybrid representation is that the iconic primitives provide a starting place for description but do not limit the user. The user is able to appropriate icons for particular contextual meanings which Mnemosyne can make some effort to interpret by relation to related contexts.

The archive is used for automatically retrieving and composing entries; one user of Media Streams is the IDIC system [Davis and Sack, 1993] which automatically composes new "Star Trek" trailers out of pieces of existing ones. Work in progress

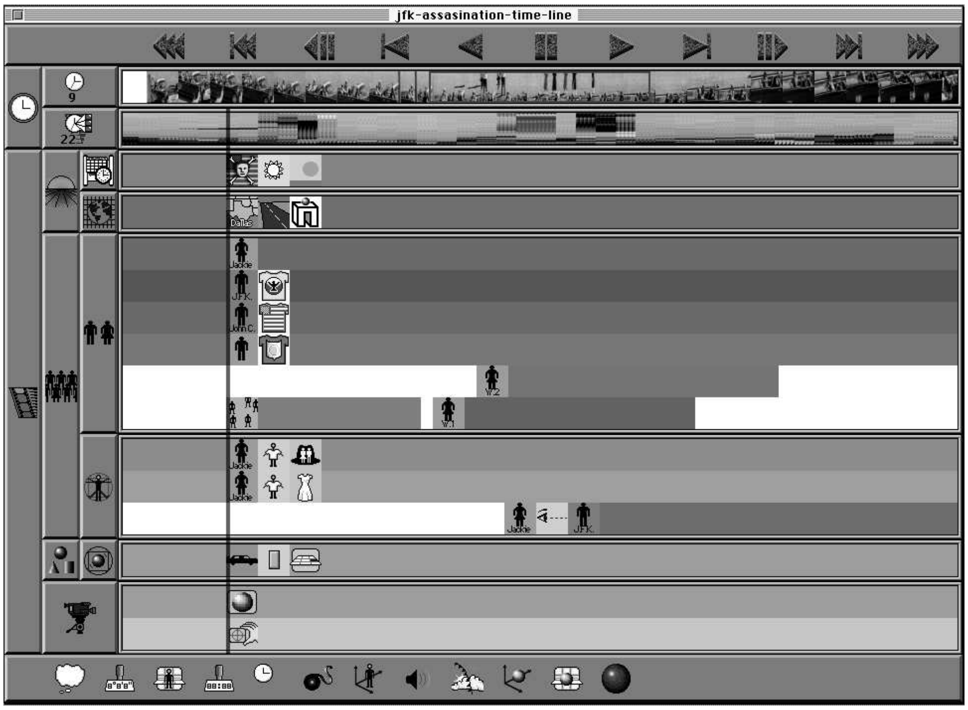


Figure 8: One component of Media Streams is the the 'Media Time Line' which is used for displaying and placing content markers (described in FRAMER) in a stream of video information.

is moving towards example-based generation which makes use of Mnemosyne to find video segments which make sense both individually and contextually.

Another possibility for this work is to use the Media Streams logs — which describe physical motion and interaction in some detail — to povide some of the vital 'common sense knowledge' that human beings learn through embodied experience and which no amount of automatic text processing will ever be able to generate.

In [Chakravarthy *et al.*, 1992], Mnemosyne was used to maintain a small library of graphical images indexed by their geometric descriptions. Retrieval from this archive was done by analogical matching between descriptions. Images were matched analogically (much as by Gerlenter's early geometric analogy program) and match failures constructed a difference network for indexing (Figure 9).

9 Conclusions and Future Work

Analogical representation reuses past analogical reasoning to support case based reasoning in the present. This reuse allows for an radically example-based account of representation where matching knowledge is built up by past analogical problem

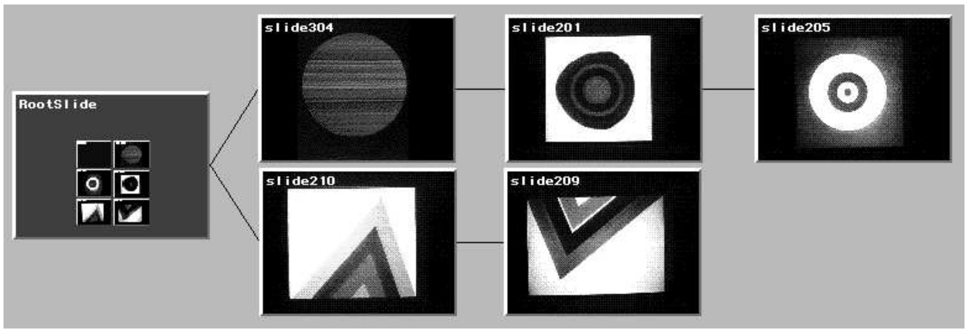


Figure 9: An image archiving system built in FRAMER drew analogies between image descriptions and used the similarities and dissimilarities to form a difference network for indexing.

solving. Such an account of rpresentation is attractive as a cognitive model due to its parsimony: it avoids positing special mental structures for the explicit description of abstractions. It is attractive as a practical approach to knowledge accquisition because it includes the invention of appropriate representations (based on exemplars) as integral to the accquistion process.

The reuse of past analogy in current matching was demonstrated with a language learning example; based on a series of examples, the system accquired the ability to identify 'meaning equivalence' in sentences based on (at first) systematic analogies and (eventually) the record of past analogies left in memory (both as prototype relations and explicit indexing traces). In addition to using conventional matching techniques, two new mechanisms were introduced: cognate matching uses a quickly computed relation between descriptive elements which computes new analogies between elements based on their indexing to common prototypes; trace matching looks at records of past expensive matches to determine new matches in a kind of 'explanation-based analogy'.

Our group's current work is addressed to building up a large storybase of short texts (one-paragraph news summaries) matched against one another (with human oversight) to build up an example based representation of aticle content. Based on this corpus, we hope to explore issues of astract and differential summarization, information retreival based on analogical encoding, and the general issues of scalability in an analogical representation. While we used language learning as an example in this paper, we believe that additional mechanisms (particularly having to do with levels of granularity) are neccessary for practical language understanding. To build up our storybase, we are a rule-based parser to generate (ambiguous) modification graphs and using these for matching and indexing.

10 References

[Agre and Chapman, 1987] Phillip E. Agre and David Chapman. Pengi: An implementation of a theory of activity. In *Proceedings of AAAI-87*. American Association for Artificial Intelligence, 1987.

[Agre, 1988] Phillip E. Agre. *The Dynamic Structure of Everyday Life*. PhD thesis, Massachusetts Institute of Technology, 1988.

[Bareiss, 1989] Ray Bareiss. *Exemplar based knowledge accquisition: A unified Approach to Concept Representation, Classification, and Learning*. Academic Press, 1989.

[Chakravarthy *et al.*, 1992] Anil Chakravarthy, Louis Weitzman, and Kenneth B. Haase. A Uniform Memory-based Representation for Visual Languages. In *ECAI-92*. European Conference on Artificial Intelligence, Springer-Verlag, 1992.

[Chapman, 1991] David Chapman. *Vision, Instruction, and Action*. MIT Press, 1991.

[Davis and Sack, 1993] Marc E. Davis and Warren Sack. IDIC: Assembling Video Sequences from Story Plans and Content Annotations. (Unpublished Working Paper, MIT Media Laboratory), 1993.

[Davis, 1991] Marc E. Davis. Director's Workshop: Semantic Video Logging with Intelligent Icons. In *Notes from Workshop on Intelligent Multimedia Interfaces*. American Association for Artificial Intelligence, 1991.

[Dreyfus, 1979] Hubert Dreyfus. *What Computers Can't Do (2^{nd} edition)*. Harper and Row, 1979.

[Falkenhainer *et al.*, 1989] Brian Falkenhainer, Kenneth D. Forbus, and Dedre Gentner. The structure-mapping engine: Algorithm and examples. *Artificial Intelligence*, 41:1–63, 1989.

[Gentner, 1983] Dedre Gentner. Structure Mapping: A Theoretical Framework for Analogy. *Cognitive Science*, 7(2), 1983.

[Haase, 1993] Kenneth B. Haase. FRAMER: A Portable Persistent Representation Library. In *Proceedings of the AAAI Workshop on*. American Association for Artificial Intelligence, 1993. (Submitted).

[Laird *et al.*, 1986] John E. Laird, Paul S. Rosenbloom, and Allen Newell. Chunking in soar: The anatomy of a general learning mechanism. *Machine Learning*, 1(1), 1986.

[Lakoff, 1987] George Lakoff. *Women, Fire, and Dangerous Things: What Categories Reveal about the Mind.* University of Chicago Press, 1987.

[Langley and Thompson, 1991] Pat Langley and Kevin Thompson. Concept Formation in Structure Domains. In *Concept Formation: Knowledge and Experience in Unsupervised Learning.* Morgan Kaufmann, 1991.

[Lenat and Guha, 1990] Douglas B. Lenat and L Guha. *CYC: Building Large Knowledge-Based Systems.* Digital Press, 1990.

[Mel̕čuk, 1988] I.A. Mel̕čuk. *Dependency Syntax: Theory and Practice.* State University of New York Press, 1988.

[Minsky, 1977] Marvin Minsky. Plain talk about neurodevelopmental epsitemology. In *Proceedings of the 5th International Joint Conference on Artificial Intelligence.* IJCAI-77, 1977.

[Minsky, 1986] Marvin Minsky. *The Society of Mind.* Simon and Schuster, 1986.

[Rosch, 1973] Eleanor Rosch. On the internal structure of perceptual and semantic categories. In T. Moore, editor, *Cognitive Development and the Acquiestion of Language.* Academic Press, New York, 1973.

[Schank and Riesbeck, 1989] Roger Schank and Chris Riesbeck. *Inside Case-Based Reasoning.* Lawrence Earlbaum Associates, 1989.

[Schank, 1982] Roger Schank. *Dynamic Memory: A Theory of Learning in People and Computers.* Cambridge University Press, 1982.

[Winograd and Flores, 1986] Terry Winograd and Fernando Flores. *Understanding Computers and Cognition: A New Foundation for Design.* Addison-Wesley, 1986.

[Winston, 1979] Patrick H. Winston. Learning and reasoning by analogy. *Communcations of the ACM*, 23(12), 1979.