



MARVIN: A Web-Based System for Representing, Retrieving, and Visualizing Analogies

HARRY J. FOXWELL and DANIEL A. MENASCÉ

{hfoxwell,menasce}@cs.gmu.edu

Department of Computer Science, MS 4A5, George Mason University, 4400 University Dr., Fairfax, VA 22030, USA

Abstract

Analogies are essential in human cognition, reasoning, learning, communication, and problem solving. They can have a profound and broad effect on how we view and understand our world. In this paper we discuss the design, implementation, and evaluation of MARVIN (Markup for Analogy Representation and Visualization for the InterNet), a Web-based system for representing, retrieving, and visualizing human-conceived analogies that provides a medium and a common language for analogy practitioners to share their analogies. We developed a compact XML content model for analogy expressions for use in Web-based environments, and show that the model is capable of representing a wide range of human-conceived analogies. We demonstrate, using XSLT, several example methods for visualizing analogy expressions that use our model. We demonstrate methods for storing and retrieving such expressions and for ranking the retrieved expressions. We designed and implemented MARVIN to demonstrate these methods. A formative evaluation of the MARVIN system found that its visualization and retrieval capabilities are of value to analogy authors and end users. A performance test showed that MARVIN's analogy retrieval is scalable to large analogy archives.

Keywords: analogies, XML, XSLT, visualization

1. Introduction

Analogies pervade all human communication and learning. They occur in an extraordinary scope and variety, ranging from the simplest *ratio* form, such as “hand is to arm as foot is to leg,” through extended analogical essays proposing that a computer is like a brain [67], the Internet economy is like the England railroad boom of the 1800s [6], and the mind of an autistic is like a Web browser [32].

Analogies are widely used when explaining ideas, especially in instructional contexts. Using analogies is “one of the core processes of cognition” [18], and may be the *primary* process of all cognition and communication [36]. Analogies are a key component of learning-by-example, or case-based reasoning [60], and are quite common in science education [30,51]. And, although analogies are generally imperfect and may lead to incorrect mappings or conclusions, their power and utility come from their ability to quickly convey large amounts of information [17] and to connect to their readers’ background knowledge and past experiences.

An analogy describes a *perception* that two different objects or situations are similar at some abstract level. Because they are perceptions, analogies are highly personal and

contextual. That is, what one person perceives and proposes as an analogy may or may not be understandable to another person, depending on its context and on the receiver's background knowledge. Analogies assist in acquiring new knowledge by attempting to map the *structure* of existing knowledge to new situations [25]. For example, in a widely known and often cited analogy, Ernest Rutherford in 1910 proposed that an unfamiliar idea – the structure of the atom, is *like* the structure of a presumably familiar object – the solar system [7,28]. In this example, the atom is the *target* analog of the analogy and the solar system is the *source* analog. This analogy suggests that we try to *map* what we currently know about the source – the solar system's parts and their interrelations, to the target – the parts of the atom, allowing us to make plausible inferences and predictions and to form hypotheses about the target object. Some of these predictions and inferences may turn out to be wrong, but the analogy provides a useful structure, or *scaffold*, for generating and considering them [8,54].

We suggest the need for a mechanism to represent, record, share, and retrieve human-conceived analogies in a structured and globally accessible form, for communities of analogy practitioners. Such global communities, based on the Web, are already forming [57]. The Web's growth and scope is an important enabler of the formation of communities of practice centered on personal, social, and especially professional interests. The Web provides the medium for communication, and XML supports the development of a common language for each community.

We therefore require a compact and general representation for analogies that supports their expression and visualization using standard Web-based technologies, and that is relatively easy to understand, author, and extend. Our approach is to provide an augmentation of existing Web documents that contain analogies rather than embedding contextual markup within those documents. This approach recognizes the immense volume of existing HTML-based Web content that will persist for many years, and provides a mechanism for content authors to include easily created structured contextual information about analogies described in their Web documents.

This paper describes MARVIN (Markup for Analogy Representation and Visualization for the InterNet), a system that defines an XML-based representation for analogies and demonstrates how it can be used to represent and visualize analogy expressions, and to store, retrieve, and share such expressions in local and Web-based archives.

We need a compact and intuitive representation of analogies that is capable of expressing much of the range of analogies that humans are able to generate. Such an expression must be consistent with current theory and research findings on analogy components and structure, and must be programmatically useful to permit Web-based sharing, storage, retrieval, and manipulation of the expressions. The MARVIN system defines such an XML content model for the creation of analogy representations; with this model, analogy authors can create analogy expressions using editors that create XML files.

We also seek a method for producing multiple visualization forms from our analogy expression that separates the process of producing the visualization from the expression of the content and structure of the analogy. Visualizations are cognitive aids that promote memory and information processing [66]; visualizations of analogies facilitate learning and enhance the use of analogies in both instruction and problem solving [13,51]. Under-

standing of the analogy is significantly improved when the relational structures present in the analogy can be visualized in a tabular or graphical form [45,51]. The MARVIN system provides this capability through the use of XSLT stylesheets.

For those who wish to use analogies for instructional or explanatory purposes, a searchable archive of analogy expressions may be queried to locate an appropriate analogy for the topic under discussion. Or, if a proposed analogy is not understood by the learner or reader, the archive may be queried to locate additional or alternate analogies better suited to the learner's background knowledge. The MARVIN system includes an analogy archive and techniques to query this archive.

The primary contribution of our work is the design and development of MARVIN. Specific contributions of our work are:

- A compact, general representation of analogy expressions, using an XML content model for universality and Web-based access. This model is powerful enough to represent analogies from varied domains, such as science, history, medicine, religion, and literature.
- Multiple visualization methods, such as tabular and graphical displays, which are generated *directly* from the XML analogy representations using the Extensible Stylesheet Language (XSL) [77].
- Web-based methods for storage, retrieval, and ranking of analogy expressions.

The above contributions use our model of the components and structure of analogies. This model is consistent with both common usage and formal characterizations of analogies from cognitive science research, from educational practice and research, and from work on Web-based knowledge representation.

The remainder of this paper is organized as follows. Section 2 provides a general overview of the MARVIN system. Section 3 reviews and defines the components of analogies, discusses the basic structure of analogies, and introduces a graphical representation for several key structures commonly found in analogies. Section 4 discusses the need for a general-purpose representation for analogies, discusses design goals for such representations, presents a formal definition for analogy expressions, and implements that definition using an XML content model. The next section discusses how analogies that are expressed using our XML content model may be visualized in several forms using Web-based tools such as XSL Transformations (XSLT) [77] and Scalable Vector Graphics (SVG) [70], and discusses design goals for such visualizations. Section 6 discusses the storage and retrieval of analogy expressions, and introduces and demonstrates methods for ranking the results of queries of analogy expression archives. Section 7 presents our proof-of-concept prototype of the MARVIN system. Section 8 presents the results of a formative evaluation of the MARVIN system by content authors and individual users, as well as the results of a scalability analysis of MARVIN's retrieval engine. Finally, Section 9 summarizes our main contributions and discusses future research suggested by this work in the area of analogy representation, visualization, and retrieval.

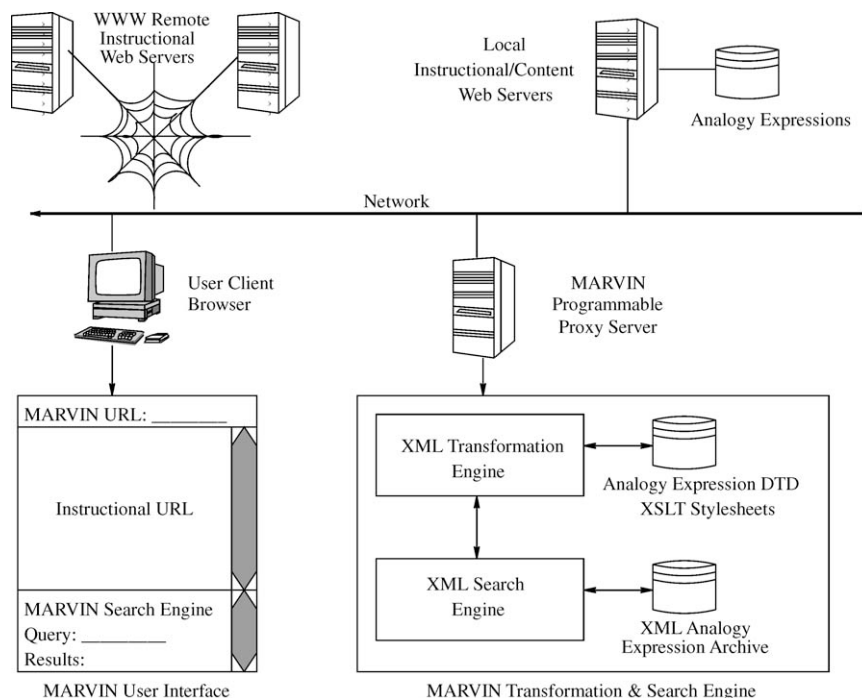


Figure 1. The MARVIN system architecture.

2. General overview of the MARVIN system

Analogy research historically has focused on several basic and overlapping areas – understanding the cognitive processes of analogical reasoning [16,26,35,43], computer simulation of human analogical reasoning [19,22,26], and using analogies in educational settings [30,51,60]. The work presented here is focused on helping humans *share* and *use* analogies they have already conceived or discovered, rather than using computers to discover analogies or to perform analogical reasoning.

Human analogical reasoning typically involves the following processes: (a) recall a source analog, (b) map the components of the source to the target, (c) generate plausible inferences about the target, (d) evaluate the inferences about the target, and (e) accept (learn/remember) new knowledge about the target [37]. MARVIN focuses on helping humans with the first two steps – retrieving source analogs already perceived and described by others and visualizing the mappings between the source and target analogs, and the final step – remembering the analogy. Generating and evaluating inferences suggested by the analogy remains the responsibility of the analogy author and users. That is, we are not concerned here with computer-aided analogy generation or reasoning, although MARVIN could be integrated with such systems.

MARVIN (see Figure 1) supports representation and visualization of *human-conceived* analogies in a common format accessible to any Web browser. It is designed to help anal-

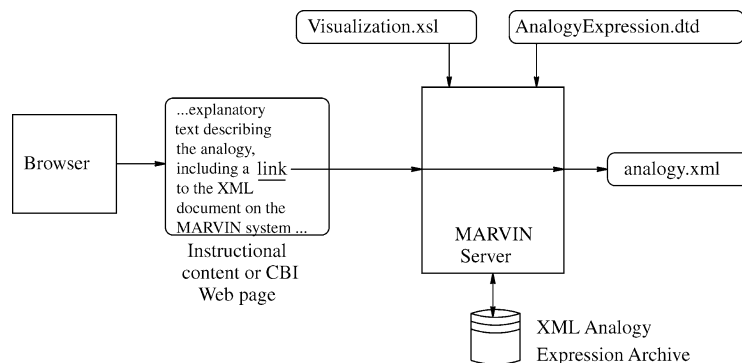


Figure 2. Integrating MARVIN into an instructional or CBI Web page.

ogy practitioners, such as authors and teachers who use analogies in instructional Web content, to represent and display the *concepts* and *relations* contained in an analogy. Additionally, it helps end users of the system to display *tabular*, *graphical*, and other forms of visualization of the analogy, enhancing their understanding of the target analog. MARVIN also enables users to *query* an analogy *archive* for analogy suggestions and to obtain *alternatives* to the analogy proposed by the content author.

Implementing MARVIN using standard and emerging Web technologies ensures global accessibility through any HTML-capable browser, and allows developers who wish to use MARVIN analogy expressions to use widely available programming tools to produce new types of visualizations. The use of XML to describe analogies allows authors to represent the essential components of the analogies independent of their visualization. This enables programmers to create many forms of visualization from the same representation, using XML transformation technologies such as XSLT stylesheets.

Although MARVIN does not require integration with instructional Web systems, it can be used as a companion to such systems. Authors of any Web content that uses analogies can create and link to a MARVIN analogy expression to augment the explanations of the analogies. Referencing the link in this way allows the user to read about the analogy in the author's primary Web page, and then follow the link to the MARVIN expression to obtain visualizations of the analogy or look up alternate analogies for the same target. Figure 2 shows this process – the user follows the analogy link in the main content page, and the MARVIN system references the analogy expression and the MARVIN DTD (i.e., *AnalogyExpression.dtd*) along with the appropriate visualization stylesheet. The user can also query the analogy expression archive to obtain additional analogies referring to the presented target analog.

Content authors should first identify an analogy they wish to represent using the MARVIN system, and then proceed as follows: (i) identify the analogy's concepts, properties, and relational structures; (ii) create a MARVIN analogy expression using an XML editor or the MARVIN Analogy Editor [56]; (iii) upload the analogy expression to the MARVIN analogy archive; and (iv) create a link to the analogy expression on their main content Web page. MARVIN users then access the expression through the MARVIN Web site, which

provides various visualizations of the analogy and the ability to look up alternatives to the original analogy.

The MARVIN system (see Figure 1) is designed to use a programmable proxy server that can process Web-based Analogy Expression documents as well as those stored in the MARVIN archive. The proxy server runs an XML transformation servlet that processes the XML Analogy Expression, using an XSL stylesheet; when the user selects a specific type of visualization, the appropriate stylesheet is called. The stylesheet transforms the XML document into HTML for a tabular visualization of an analogy, or into SVG for a graphical visualization.

3. The structure and components of analogies

We include here several examples from the analogy literature to illustrate and develop a terminology for the structure and components analogies. We revisit these examples in Sections 4, 5, and 6 to illustrate representation, visualization, and retrieval, respectively. Additional examples may be found on the author's Web site [21].

Gentner [68] emphasizes the importance of relations in analogies and states: "... an analogy is a mapping of knowledge from one domain (the base) into another (the target), which conveys that a system of relations that holds among the base objects also holds among the target objects ... in interpreting an analogy, people seek to put the objects of the base in one-to-one correspondence with the objects in the target ...".

The *objects* in the above definition may be words, sounds, images, processes, or other symbols representing perceived concepts and the relations among them. In this paper, we define an analogy as a set of proposed similarity mappings between an unknown set of concepts and relations (the *target* analog) and a known set of concepts and relations (the *source* analog), used for instructional or explanatory purposes.

Our operational definition of a concept is the *word* or *words* a content author chooses to describe the abstraction conveyed by a concept. This is consistent with the view that a concept is a mental representation and that a word is an expression of that representation [49]. The base form of the word can then be used as a key to interface with lexical databases, such as WordNet [47], or with conceptual graph tools such as WebKB [44], in order to obtain a better understanding of the meaning and context for the concept word. Similarly, we represent a relation as a word chosen by an analogy author to represent an association between two analogy components.

Analogies are perceived correspondence *maps* between the respective components and structures of a familiar source analog and an unfamiliar target analog. There are three ways in which the perceptions of objects or collections of objects may be placed in correspondence [10]. The simplest type of perceived correspondence is that of properties (e.g., same color, shape, or size). Another type of perceived correspondence is that of concepts (e.g., *nucleus* maps to *sun* in Rutherford's analogy). The third type of perceived correspondence is *system correspondence*, also called *systematicity* [25]. In this type of correspondence, the relational structures among objects in the target analog are mapped to similar or identi-

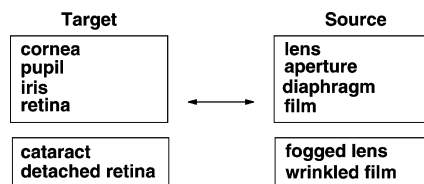


Figure 3. The eye/camera analogy.

cal relational structures in the source analog (e.g., the electron is hypothesized to *orbit* the nucleus as a planet *revolves* around the sun).

We use these correspondence types to define an *analogy expression* as a collection of *correspondence maps*, which associate the concepts, properties, and relations of the target with those of the source. The systematicity principle additionally specifies that analogies must preserve *higher order relations* such as causality and implication when mapping the components of the source to those of the target. Using the Rutherford analogy, for example, we can then form (and test) a hypothesis that electrical attraction *causes* the electron to orbit the nucleus in the same way that gravitational attraction *causes* the planet to revolve around the sun [73]. That is, the proposed analogy preserves in the target the higher order *causal* relationship perceived in the source.

3.1. Analogy examples

In addition to the Rutherford analogy discussed above, we now examine several additional examples to illustrate the types of analogy structures that can occur. Note that our goal is to *describe* human-conceived analogies without attempting to evaluate their correctness or completeness.

The Altoona List of Medical Analogies [57] lists a simple analogy comparing the eye to a camera in order to explain certain types of vision problems to patients (see Figure 3). It first establishes a mapping of the parts of the eye to those of a camera, and then explains that a cataract in the eye is like a fogged lens in a camera. Continuing with the analogy, it explains that a detached retina is like a camera with wrinkled film. Once this analogy is established, the eye doctor can continue, perhaps with additional analogies, explaining the necessary procedures for treating the conditions. This analogy is primarily a mapping of known concepts – camera parts and their presumably understood functions, to unfamiliar concepts – eye anatomy and vision impairments.

Upon observing the moons of Jupiter, Galileo formed an analogy between the solar system and the Jovian system, proposing that the *relationship* of the planets to the Sun was the same as that of the moons to Jupiter [23] – that the planets, *including the Earth*, orbit the Sun like the moons orbit Jupiter. This comparison illustrates a fundamental characteristic of analogies – the mapping of *relations*, as shown in Figure 4.

Analogies that map higher order relations such as causality can provide significant insight into the target analog. In the previously discussed Rutherford analogy, gravitational force as a *cause* of the planets' motion around the Sun is mapped to electrical force as a

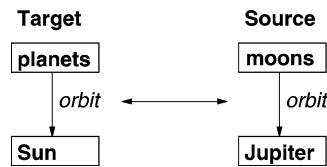


Figure 4. Galileo's solar system analogy.

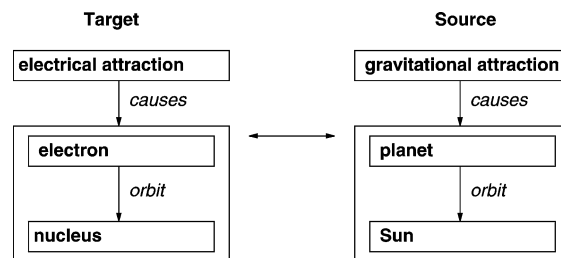


Figure 5. The Rutherford analogy.

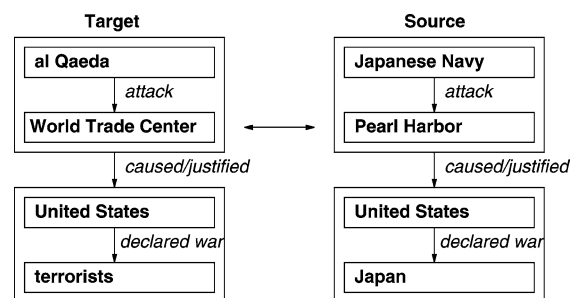


Figure 6. Historical analogy: September 11 = Pearl Harbor.

hypothesized *cause* of electrons “orbiting” the nucleus. In this analogy, a source concept is therefore associated with a source relation, and a similar relational structure is proposed in the target analog, as shown in Figure 5.

Historical analogies can be created and used by politicians and journalists to sway public opinion or to suggest the inevitability of a decision or course of action. The September 11, 2001 attack on the World Trade Center and Pentagon has been compared to the Japanese Navy’s attack on Pearl Harbor in 1941 [12]. Without evaluating the merits of the analogy, we see that the analogy can be partially expressed as a higher order relational map. The target consists of an action (al Qaeda attacks WTC) causing (or implying a desired decision) an action (US declares war on terrorists); the source is similarly structured – (Japanese Navy attacks Pearl Harbor) causing (or implying a desired decision) an action (US declares war on Japan). Figure 6 illustrates this map.

We observe that although analogies may be quite elaborate or complex [6,32], they may be decomposed into structured components such as those illustrated above, each of which

maps target analog components to identical component types in the source analog. A more formal definition of these structured component types is as follows.

Definition 1 (ConceptSet). A *ConceptSet* $C = \{c_1, \dots, c_n\}$ is an ordered set of one or more concepts. Note that the order of the elements listed in a *ConceptSet* is significant since we map corresponding concepts in the target and source. In the Rutherford analogy, for example, a possible target *ConceptSet* is {electron, nucleus}, and a corresponding source *ConceptSet* is {planet, sun}.

Definition 2 (PrimaryRelationStructure). A *PrimaryRelationStructure* $P = (C_a, R, C_b)$ associates *ConceptSets* C_a and C_b through zero or more relations in the list of relations R . We say “zero or more” because relations can be implied in an analogy rather than explicitly named.

In its most common form, a *PrimaryRelationStructure* consists of two concepts associated by a single relation as in ({planet}, revolves, {sun}). Note that there may be multiple relations between the *ConceptSets* (e.g., sun is *larger than* a planet, sun is *more massive than* a planet, sun is *hotter than* a planet, etc.), and that there may be more than one element in each concept set (the concept set containing the single element *planet* could be replaced by the set {Mercury, Venus, Earth, Mars, Jupiter, Saturn, Uranus, Neptune, Pluto}, for example).

Definition 3 (ConceptToRelationStructure). A *ConceptToRelationStructure* is a tuple of the form (C, R, P) where C is a *ConceptSet* associated by zero or more relations in the list R to the *PrimaryRelationStructure* P .

Definition 4 (RelationToConceptStructure). A *RelationToConceptStructure* is a tuple of the form (P, R, C) where P is a *PrimaryRelationStructure* associated by zero or more relations in the list R to the *ConceptSet* C .

A *RelationToConceptStructure* and a *ConceptToRelationStructure* associates relations with concepts or concepts with relations, respectively, depending on the directionality of the higher order relation being specified. Higher order relations can include causality, implication, and sequencing, for example.

Definition 5 (RelationToRelationStructure). A *RelationToRelationStructure* is a tuple of the form (P, R, P) that associates a *PrimaryRelationStructure* to another *PrimaryRelationStructure* through zero or more relations in the list R .

There are thus five types of maps, each of which maps a structure (e.g., *PrimaryRelationStructure*) in the target analog to a structure of the *same type* in the source analog: Figures 7 and 8 illustrate the five types of maps discussed above.

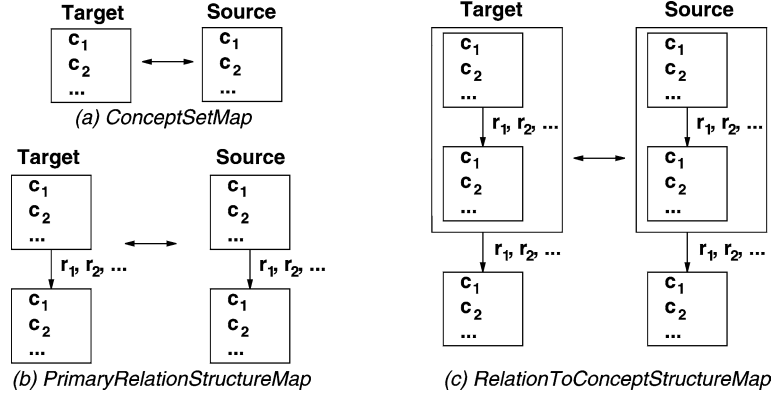


Figure 7. Maps in analogies: (a) *ConceptSetMap*; (b) *PrimaryRelationStructureMap*; (c) *RelationToConceptStructureMap*.

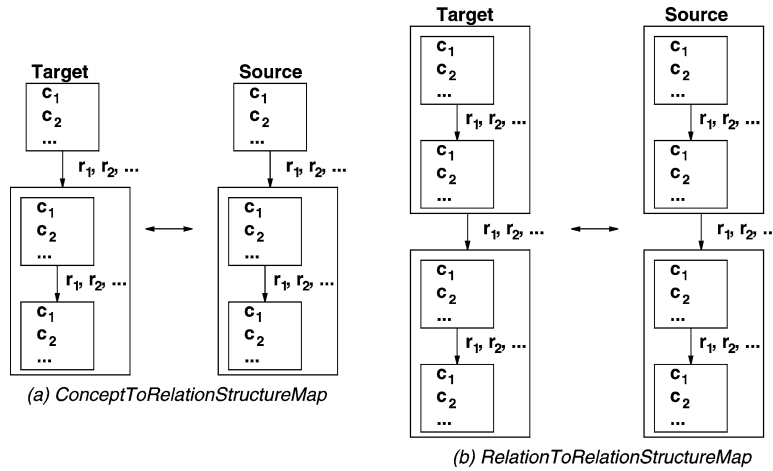


Figure 8. Maps in analogies: (a) *ConceptToRelationStructureMap*; (b) *RelationToRelationStructureMap*.

Consider the analogy that compares the human circulatory system to home plumbing, used by physicians to explain cardiovascular diseases to medical patients [58] (see Figure 9). The target concepts *blood*, *blood vessels*, and *blood pressure* are mapped to the source concepts *water*, *pipes*, and *water pressure*, respectively, forming a *ConceptSetMap*. When explaining congestive heart failure to a patient, the physician first describes the similarity of the concepts of the circulatory system to those of a home plumbing system, implicitly using the *ConceptSetMap*, and then describes how the concepts are related, creating and explaining relational structures and maps. For example, the physician explains that a clogged artery can *cause* the blood to back up into the lungs similar to the way that a clogged drain can *cause* the water to back up and overflow, thereby mapping the causal relation of the source analog to that of the target. Thus, we see that this analogy, as it is

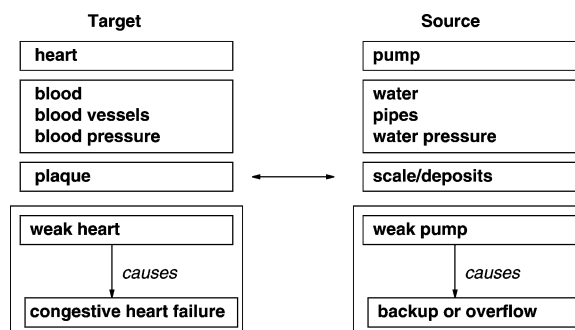


Figure 9. Circulatory system analogy.

presented by its author, may be decomposed into several ConceptSetMaps and a PrimaryRelationStructureMap. Note that we can create multiple ConceptSetMaps, grouped by type of concept, as shown in Figure 9. This analogy and its MARVIN visualization may be found on the author's Web site [21], along with additional analogy examples taken from a variety of knowledge domains.

3.2. The limitations of analogies

Analogies are like chainsaws – powerful and useful tools that can injure you if you misuse them. Like incorrect maps, bad analogies can lead you away from your destination or cause you to become entirely lost. The over-reliance on limited analogical models can impair discovery and the formation of useful hypotheses. The early history of science includes many instances of misleading and inaccurate analogies. Alchemists, for example, were constrained in their understanding of matter and chemistry by their adherence to analogies between elements and animal or human characteristics [27].

Practitioners of analogical explanations recognize the dangers of misleading analogies. A doctor who frequently uses analogies to explain medical concepts notes that his patients can transfer inappropriate knowledge from the source to the target of an analogy, and suggests that good analogies should (i) be visual, (ii) illustrate the necessary concepts, (iii) use a familiar source, and (iv) be clear and short [58]. Additionally, those explaining or teaching an idea should consider whether an analogy is even necessary. Analogies should be used primarily when the idea being taught is new and is hard for the learner to understand. The analogy's limitations should be discussed, and dependence on the analogy reduced as the learner progresses in understanding the target.

Analogy practitioners do indeed recognize the limitations of analogies, and usually include a final recommended step in the analogical reasoning process – to indicate where the analogy breaks down [29,42], although they note that this should generally be done by the proposer of the analogy. This suggests that the process of *evaluating* the analogy is *separate* from the *depiction* of the analogy itself, as we discuss in Section 4.

In spite of their dangers and misuses, analogies remain an important and widely used tool for communication and learning. Our research focuses on how to represent and visu-

alize some of the enormous range of human-created analogies. The final evaluation of an analogy's usefulness, however, is the responsibility of its proposer and users.

4. Representing analogies in XML

Here we use the characterization of analogies presented in Section 3 to create an XML content model [74] capable of describing a wide variety of analogies. The design of such a model for analogies must meet several general criteria and must also meet the needs of content authors who select or create analogies, content readers who learn using the analogies, and program developers who use the model to create new ways to use the analogies.

We distinguish between an *analogy* – the human's perception of sameness, and an *analogy expression* – the representation of that perception. The representation is necessarily a limited and imperfect model of the complete human perception, however. Our goal is to define a symbolic representation of analogies that captures a wide scope and variety of human-conceived analogies. Specifically, it must be able to express both simple similarity of properties and concepts, and must preserve relational structures that are the core of analogy perceptions [25]. Because analogies are remembered, reused, and extended over time to generate new inferences [36,40], our representation must also be extendable. That is, it must be flexible enough to permit the addition of new analogy components as needed by the author, user, or developer.

Authors of instructional or explanatory content frequently use analogies to assist the student or reader in understanding new ideas. An analogy representation must therefore be compact yet expressive enough to allow the author to record the essential components of the analogy, and must also permit some indication of the validity of the proposed comparisons. In our analogy expression, we therefore use simple words and short phrases as expression "primitives" along with expressions that describe the structure of the relations among the primitives.

We want to provide access to analogy expressions using familiar, Web-based tools and technologies such as browsers. Moreover, we need to separate the representation of the structure of the analogy expression from its visualization. XML was designed specifically for the purpose of separating representation from expression [74], allowing multiple forms of visualization based on a common representation model.

Our representation is implemented as an XML DTD (Document Type Definition), permitting a compact, common format for describing, searching, and transforming analogy expressions. Knowledge representations defined using XML may be expressed using either DTDs or schemas. Like XML DTDs, XML Schemas are used to define the structure, content, and semantics of XML documents [76]. Schemas provide for strong data typing and validation, explicit cardinality controls, and constraints on attribute values. But for representations that do not require strong data typing or cardinality controls, DTDs are sufficient and simpler [46]. Tools for creating, validating, and transforming DTD-based XML files are mature and widely available [11]. Because the analogy expressions discussed here consist exclusively of a relatively small number of text-based elements with *any number*

of repeated components, a DTD was developed. XML analogy expressions based on our DTD may be transformed as needed into other forms using XSL stylesheets.

4.1. *Analogy expression DTD elements*

The primary components of analogy expressions are the words used to describe concepts and relations, and these are represented by elements in the DTD. Words describing properties of concepts are also represented as elements because the words themselves may have attributes. The words selected are those explicitly used or implied in the narrative description of the analogy. The Analogy Expression DTD defines the Analogy element as

```
<!ELEMENT Analogy (TargetDescription, SourceDescription, Map*)>
<!ATTLIST Analogy AnalogyName CDATA #IMPLIED Reference CDATA #IMPLIED >
<!ELEMENT TargetDescription (#PCDATA)>
<!ATTLIST TargetDescription Domain CDATA #IMPLIED >
<!ELEMENT SourceDescription (#PCDATA)>
<!ATTLIST SourceDescription Domain CDATA #IMPLIED >
```

That is, an analogy expression consists of a pair of text descriptions, followed by zero or more Map elements. The mapped elements are modeled separately from the descriptions to emphasize the mappings of the structures components of an analogy. The Analogy attributes include AnalogyName for indicating a general title or name for the analogy, and Reference for indicating the source of the analogy. The Reference attribute may include a URL. The description elements are brief phrases describing the target and source analogs. By convention, target elements in the expression are always defined first. A Domain attribute is included for both the target and for the source to indicate the general knowledge domains being referenced in the analogy. All elements modeled as words or phrases use the #PCDATA (untagged character data) type. For example, the start of the Rutherford analogy expression can be tagged as

```
<Analogy AnalogyName="Rutherford Analogy"
      Reference="Bohr, Niels. Nobel Lecture, December 11, 1922,
      http://www.nobel.se/physics/laureates/1922/bohr-lecture.html">
  <TargetDescription Domain="Physics">atomic structure</TargetDescription>
  <SourceDescription Domain="Astronomy">solar system</SourceDescription>
  ...
</Analogy>
```

The “primitive” elements of an analogy expression are concepts, properties, and relations, defined using #PCDATA as follows:

```
<!ELEMENT ConceptDescription (Concept, Property*)>
<!ELEMENT Concept (#PCDATA)>
<!ELEMENT Property (#PCDATA)>
<!ATTLIST Property Validity (high|low|none) #IMPLIED >
<!ELEMENT Relation (#PCDATA)>
```

We include an optional *validity* attribute for Properties to allow for expression of concept properties that may not be relevant to the analogy. For example, in the Rutherford analogy, *hot* may be specified as a property of the sun, but there is no corresponding relevant temperature property for the nucleus. Thus, if the temperature property for the sun concept is represented in the expression, the analogy expression author can label it with a validity indicator of “none,” while the mass property “heavy” can be labeled with a validity indicator of “high” since the relative masses of the sun/planet and nucleus/electron are essential to the analogy. This can be tagged as

```
<ConceptDescription> <Concept>sun</Concept>
  <Property Validity="high">heavy</Property> <Property Validity="none">hot</Property>
</ConceptDescription>
```

The author of the expression can simply omit irrelevant properties.

In designing the DTD we stress the distinction between the target analog components and the source analog components by giving them separate but similar names. We also observe that in order to map only identical analogy structures, we must enumerate the possible structures rather than using more compact or recursive element definitions that would allow mapping of unlike components.

Analogy expressions can contain any number of each of the five map types discussed earlier, defined in our DTD as follows:

```
<!ELEMENT Map (ConceptSetMap | PrimaryRelationStructureMap |
  ConceptToRelationStructureMap | RelationToConceptStructureMap |
  RelationToRelationStructureMap)>
```

The *ConceptSetMap* consists of a *TargetConceptSet* and a *SourceConceptSet*, each of which must contain at least one concept description, listed in the order that pairs like concepts of the target and the source. We include a *MapLabel* attribute for each type of map to allow the author to assign a generic name to the type of concepts or relations being compared in the analogy:

```
<!ELEMENT ConceptSetMap (TargetConceptSet, SourceConceptSet)>
<!ATTLIST ConceptSetMap MapLabel CDATA #IMPLIED >
<!ELEMENT TargetConceptSet (ConceptDescription+)>
<!ELEMENT SourceConceptSet (ConceptDescription+)>
```

The *PrimaryRelationStructure*, used in the definition of the remaining map structures, consists of two *ConceptSets* associated with zero or more *Relations*:

```
<!ELEMENT PrimaryRelationStructure (ConceptSet, Relation*, ConceptSet)>
```

The remaining maps that model the structures discussed in Section 3 are defined as follows:

```
<!ELEMENT PrimaryRelationStructureMap (TargetPrimaryRelationStructure,
  SourcePrimaryRelationStructure)>
<!ATTLIST PrimaryRelationStructureMap MapLabel CDATA #IMPLIED>
<!ELEMENT ConceptToRelationStructureMap (TargetConceptToRelationStructure,
```

```

SourceConceptToRelationStructure)>
<!--ATTLIST ConceptToRelationStructureMap MapLabel CDATA #IMPLIED>
<!--ELEMENT RelationToConceptStructureMap (TargetRelationToConceptStructure,
SourceRelationToConceptStructure)>
<!--ATTLIST RelationToConceptStructureMap MapLabel CDATA #IMPLIED>
<!--ELEMENT RelationToRelationStructureMap (TargetRelationToRelationStructure,
SourceRelationToRelationStructure)>
<!--ATTLIST RelationToRelationStructureMap MapLabel CDATA #IMPLIED>

```

Note that, like the `ConceptSetMap`, each of these maps also has a `MapLabel` attribute. The appendix shows the complete analogy expression DTD.

It should be noted that there is no single “correct” model for expressing this or any other analogy. Depending on the needs of the author, the type and level of the audience, and the purpose of the analogy, the author may choose to include or exclude concepts, properties, and relations.

4.2. *Creating an Analogy Expression XML document*

A content author who proposes an analogy should first thoroughly understand the target and source analogs, along with the limitations or possible misconceptions that the analogy may create, and then describe it in narrative or diagrammatic form. Using the Analogy Expression DTD next requires the identification of each of the key words that will be selected to describe the analogy’s prominent concepts, relations, and properties that the author wishes to map.

Having identified all the key words for concepts, properties, and relations in the analogy, and having identified the proposed concept and relation maps, an XML file containing the maps can be created. A content author familiar with XML syntax can use a text editor for this purpose, but most authors will prefer an XML editor [15,48,79]. XML editors use the DTD file to define selectable element tags, simplify the markup process for the author, and ensure the creation of a valid XML file.

XML editors, such as `epcEdit` [15], display the available tags for the author to select, and show the tags used in the marked up document. However, analogy practitioners are unlikely to be skilled in understanding and creating XML expressions. For this reason, we developed an analogy expression editor [56] available on the MARVIN Web site [21], which can be downloaded and used by analogy practitioners to create MARVIN analogy expressions. The editor lists the five types of maps and displays a set of pre-structured panels for each type of map. Each map panel contains labeled text entry fields for the concept and relation components of the map. The editor is written in Java, using J2SE 1.4 [63] so that it may be installed and run on any system for which that version is available, including Windows, Mac OS X, Linux, and UNIX.

5. Visualization of analogies

Visualizations accompanying verbal or printed explanations have been shown to augment learners' internal visualizations and to enhance the development of internal visualization abilities [34]. Visualizations reduce the cognitive load on short-term memory [66], and can assist in the transfer of perceptions from the source analog to the target analog [13]. An analogy can be better understood if the relational structures of the source and the target can be visualized.

The first use of our XML analogy expression is to create visualizations of the analogy proposed by the content author. An analogy visualization is any *spatially structured* representation of the components of the analogy, showing the concepts, properties, and relations of the target analog and how they map to those of the source analog. A general method is needed that is not specific to each individual analogy expression, but one that works with *all* such expressions that use the Analogy Expression DTD. For Web-based visualizations, we need to transform the XML expressions into HTML or compatible graphical form for display in standard Web browsers. For this purpose we use XSLT, the XML Stylesheet Language for Transformations [77]. This approach emphasizes the motivation behind using XML to express analogies – separating the representation of the analogy from the type of display we wish to create. Once the analogy is represented using the DTD, then tabular, graphical, and other types of displays may be generated from the same representation and displayed on any HTML-capable output device.

We created several XSLT stylesheets to demonstrate the variety of visualization aids that may be produced. One stylesheet constructs an HTML table representation of the components and structure of any analogy expression that uses our DTD. Figure 10 shows this type of visualization of the Rutherford analogy. A modified version of this stylesheet produces HTML query strings linking the concepts in the visualization to ontology sites such as WordNet or WebKB. Other forms of visualization are possible; we developed another stylesheet that produces an SVG graphical representation of the analogy components. Figure 11 shows a graphical visualization of the Galileo example analogy expression. Additional visualizations examples may be found in [21].

A more general approach to visualizing concept associations is that of Topic Maps [24]. Although tools for displaying topic maps graphically in Web browsers have been developed [14,52], this approach has not been exploited to display analogies in a consistent representation. But because MARVIN represents analogy concepts and relations using XML, such representations can be transformed to topic maps using an XSLT stylesheet.

5.1. Visualization stylesheets

The tabular Analogy Expression XSLT stylesheet can be used for *any* analogy expression that uses the Analogy Expression DTD. This stylesheet produces an HTML document containing a table that displays the components and structure of the analogy expression, including concepts, properties, relations, and the five types of maps. The tabular visualization displays the components of the analogy, placing the target analog components on

Close Window	Target	Source
Analogy	Atomic Structure Domain = Physics	Solar System Domain = Astronomy
	<u>electrical attraction</u>	<u>gravitational attraction</u>
	<u>causes</u>	<u>causes</u>
	<u>electron</u> • small • light	<u>planet</u> • small • light
	<u>orbit</u>	<u>revolve</u>
	<u>nucleus</u> • large • heavy	<u>sun</u> • large • heavy

Figure 10. Tabular visualization of Rutherford's analogy.

the left and the corresponding source components on the right. Corresponding concepts, properties, and relations appear in the same row of the table, colors are used to highlight and organize matching components, and table cell borders of varying color and thickness are used to visually group matching components.

For a graphical rather than tabular display of the analogy expression, a stylesheet was created to produce an SVG document that can be directly interpreted by a browser via an SVG plugin [1]. Alternatively, the SVG output can be further transformed by the server into a graphic format such as GIF or JPG. The SVG visualization stylesheet is similar in structure to the tabular visualization stylesheet in that it consists of templates for each expression component that produce SVG graphics instructions instead of HTML table tag instructions. The graphical visualization displays the target analog structures on the left of the image and the corresponding source analog structures on the right. Concepts appear in rectangles, and relations are indicated using labeled lines connecting the concepts.

As discussed in Section 8, some users prefer the tabular visualization of analogies, while some prefer the graphical style; both authors and users have found such visualizations helpful in understanding analogies presented in this manner.

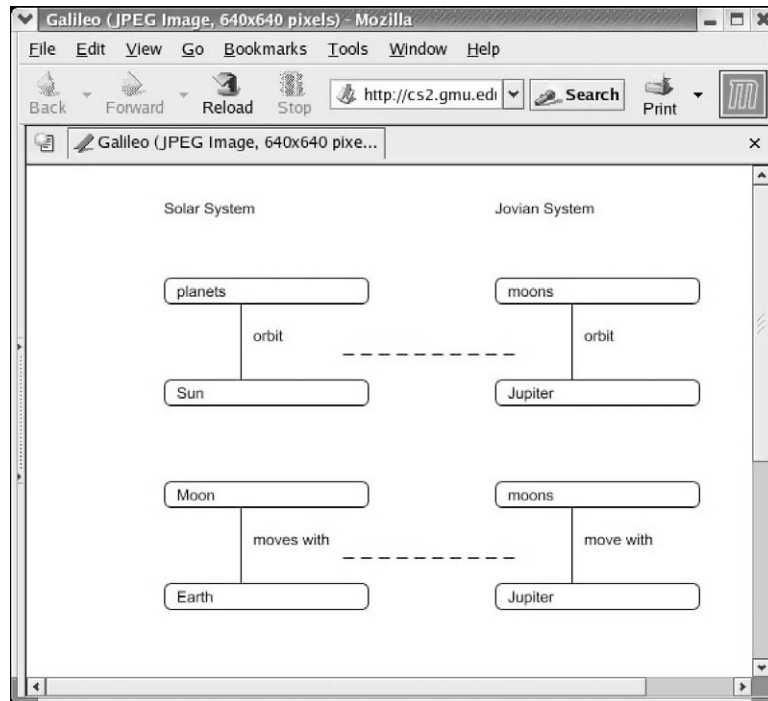


Figure 11. Graphical visualization of the Galileo analogy.

5.2. Visualization generation process

Figure 12 describes the complete sequence for creating and visualizing analogy expressions. Using the Analogy Expression DTD file and an XML editor, a content author creates an XML file for the analogy. The XSLT processor takes the XML file and stylesheet as input data, processes it using the templates in the XSLT file, and outputs an HTML document that can be viewed with a browser. Xalan [5], a separate XSLT processor, can be used to create HTML or SVG output documents from the stylesheets. The MARVIN system uses the XSLT processor embedded in Cocoon [2], along with our stylesheets, to transform XML analogy expressions into visual representations.

The HTML document produced by the XSLT processor can, of course, be edited or transformed to enhance the document with links to other Web documents or images. A key step to understanding an analogy is to understand the *meaning* of the terms used to describe it. A small modification to the XSLT stylesheet can be made to create links for each concept to an explanation or definition of that concept. Such definitions can be created explicitly, or can be extracted by querying appropriate Web resources such as Google [31], WordNet [47], or WebKB [44].

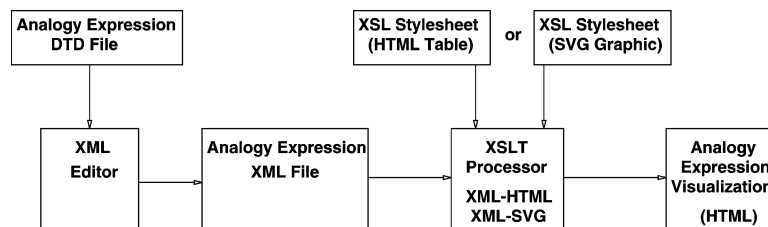


Figure 12. Process for creating and visualizing analogy.

6. Retrieving analogies

Having developed general methods for representing and visualizing analogies, we can now create archives of locally stored and Web-based XML analogy expressions. One motivation for creating such archives is to provide a mechanism for sharing creative and useful analogies among those who use them in their work. For example, a biology teacher may want to look up and evaluate analogies about the heart and circulatory system. While this is certainly possible now with search engines like Google [31] by adding “analogy” to the other search terms, a system such as MARVIN specifically stores analogy expressions in a standard XML format and provides a variety of mechanisms for structure-based retrieval and visualization of analogies, and for evaluating their pertinence to the topic under discussion.

An additional motivation for such retrievals concerns the *understanding* of proposed analogies. If we present to a student that “the eye is like a camera,” the analogy will be understood *only* if the student has background knowledge about cameras and how they work. When the student lacks such knowledge, the analogy is utterly meaningless. If, however, the student can extract from an archive *several* analogies that have the *same target* as the original analogy, he may find one or more that he understands. The presentation of multiple analogies for the same target has been shown to expand the range of hypotheses that students consider [9].

We have characterized our collection of analogy expressions as an “archive,” much like a database. Retrieving an expression from the archive in one sense is like a database *query* – any retrieval process is expected to return only “records” (analogy expressions) that match the request criteria in some manner. In database queries, there is generally no notion of *approximate* record match, either the record matches the criteria or it does not [53]. In *searches* of text documents, on the other hand, there is a notion of approximate matching and a need to order the results by some relevance metric.

However, we cannot expect all analogy expressions dealing with the same general target subject to have the same structure or even to use the same words to describe the concepts and relations. We therefore require some characterization of analogy expressions that allows us to rank one expression as “better” than another in some manner. Unlike typical text document searches, however, we are not interested in word frequency within the expression. Rather, we are primarily interested in *where* the retrieval keywords occur *within* the analogy expression structure because the type of map in which a concept word or relation

word occurs conveys additional information about the role of that word in the analogy.

We now consider the problem of retrieving analogy expressions from an archive and ordering the results according to some meaningful metric. Recall that the primary components of an analogy are concepts and relations, and that these are explicitly represented as XML elements in our analogy expressions. Note also that the main focus of interest in an analogy is the *target* analog and its component concepts and relations. We therefore require a procedure to query an archive of analogy expressions, specifically their XML Concept and Relation element contents, for expressions whose target analogs contain some form of the query keywords of interest.

It is important to emphasize at this point that for the purposes of retrieval we are not necessarily interested in the overall *structure* of an analogy expression, only that it *contains* concepts or relations in the target analog elements that are relevant in some way to the query keywords. Thus, when we query the analogy archive for alternates to a proposed analogy, we are not seeking analogical similarity in a structural sense because the alternates may have different structures, different levels of detail or complexity, or different ordering of the map elements.

6.1. Retrieval queries

We now examine two retrieval strategies for querying an archive of analogy expressions. An archive may include XML files stored in a local disk directory, and URLs for Web-based XML files. In either case, the files must conform to our Analogy Expression DTD to permit querying based on their element contents. We examine two types of queries – keyword match queries, and generalization queries.

6.1.1. Keyword match queries Recall that our analogy expressions are collections of *simple* and *compound* XML elements. *Simple* XML elements contain only parsed character data (#PCDATA); *compound* XML elements contain other elements. To illustrate, the Map element in our DTD is a compound element consisting of a ConceptSetMap or other type of map, while the Relation element is a simple element consisting of PCDATA. For our queries, we are primarily interested in the contents of the *simple* elements defined in the DTD, along with their optional attributes. These elements contain words representing concepts and relations that are structured using map elements.

The *simple* elements of interest for queries in an Analogy Expression DTD are elements of the analogy expression that are of one of the following types: (a) TargetDescription, (b) SourceDescription, (c) Concept, (d) Property, and (e) Relation. Attributes of interest for queries include: AnalogyName (of the Analogy element), Reference (of the Analogy element), Domain (of the TargetDescription and SourceDescription elements), and Validity (of the Property element).

Let A be a set (an archive) $A = \{a_1, \dots, a_n\}$ of n analogy expressions where each a_i is an analogy expression. Let T be the set of possible simple element types in an analogy expression. So, $T = \{\text{Concept}, \text{Property}, \text{Relation}, \text{TargetDescription}, \text{SourceDescription}\}$. Let $S(a_i) = \{e_{i_1}, e_{i_2}, \dots\}$ be the set of simple elements of a_i , and $t(e)$ be the element type

of simple element e , where $t(e) \in T$. Let $w(e)$ represent the contents (word or phrase) of simple element e .

Then, the result $R(A, s, q)$ of a keyword match query on keyword q associated with elements of type s is defined as the set of all analogy expressions in the archive A that contain elements of type s whose content “match” q . More formally,

$$R(A, s, q) = \{a_j \mid (a_j \in A) \wedge (\exists e \in S(a_j) \mid t(e) = s \wedge q \approx w(e))\}.$$

The notation $q \approx w(e)$ means that query keyword q “matches” the contents of element e in some sense. Such a match can include an exact character string match, case-insensitive matching, and wildcard (regular expression) matching. That is, a query for the keyword expression “atom*” should return matches for “atom,” “Atom,” and “atomic,” for example. Most text search tools permit this type of matching; the search engine used in the MARVIN system allows regular expression and case-insensitive searches.

To illustrate, a user might wish to find all analogy expressions in an archive that contain the concept *nucleus*, or the relation *orbits*, or to find all expressions in the archive whose target description domain attribute is *physics*. Examples of such queries are discussed in Section 6.3.

6.1.2. Keyword generalization queries A keyword match query will only return analogy expressions in the archive that contain the *identical* words (or regular expression approximations of the words) in the specified query elements. We may also be interested, however, in retrieving expressions that contain synonyms or other generalizations of the query keywords, because an analogy expression author may have used synonyms or more general terms to describe the analogy.

Let $g_k(w(e))$ be a set of k -level generalizations of the content word of element e . By k -level we mean the number of parent generalization steps required to obtain the set of generalizations of the query keyword. Such generalizations can include *synonyms* (words having the same or similar meaning, which are therefore 0-level generalizations), or *hypernyms* (words that are more generic than the given query keyword word and include it as an instance or example). Synonyms, hypernyms, and other k -level generalizations can be obtained manually or from online vocabulary ontologies such as WordNet [47]. For example, WordNet gives “serpent” as a synonym (0-level generalization) for the concept query keyword “snake,” “reptile” as a hypernym (1-level generalization), “vertebrate” (2-level generalization), and “animal” (3-level generalization).

Then, the result set $R_k(A, s, q)$ of a k -level generalization keyword query on keyword q associated with elements of type s is defined as the set of all analogy expressions in the archive A that contain elements of type s such that either (1) there is a non-empty intersection between the set of k -level generalizations of keyword q and the set of k -level generalizations of the contents of the elements of type s in the analogy, or (2) keyword q matches one of the synonyms of the contents of the elements of type s in the analogy. More formally,

$$R_k(A, s, q) = \{a_j \mid (a_j \in A) \wedge ((\exists e \in S(a_j) \mid (t(e) = s) \wedge ((g_k(q) \cap g_k(w(e)) \neq \emptyset) \vee (q \approx r \mid r \in g_0(w(e))))))\}.$$

Table 1. Values of the term weights.

Type of map	Type of term	
	Concept	Relation
ConceptSetMap	1	0
PrimaryRelationStructureMap	2	2
ConceptToRelationStructureMap	4	4
RelationToConceptStructureMap	4	4
RelationToRelationStructureMap	6	6

For example, we may be interested in analogy expressions containing at least one Concept element whose contents match the keyword *nucleus* or generalizations of *nucleus* such as *center* or *core*. Some search engines, such as SHOE (Simple HTML Ontology Extensions) can be configured to perform such queries [33]. The Lucene search tool used in the MARVIN system does not currently have this capability, but it is being investigated by the Lucene [3] developers. MARVIN supports the formulation of complex queries, such as Boolean combinations of simple queries.

6.2. Ordering analogy expression query result sets

Query result sets of analogy expressions should be ordered according to their relevance to the query keyword expression and according to their pertinence to the user's goal for the query. Relevance is an objective measure of closeness of a query result to the query; pertinence is a subjective assessment by the user of the usefulness of a query result [59].

When searching for analogy expressions that are related in some sense to a query containing concept or relation keywords of interest, we are primarily interested in *topic relevance* rather than any structural similarity that any two expressions may have. And, while we are interested in the pertinence of our analogy expressions, the notion of relevance as it applies to analogy expression queries needs definition beyond the usual keyword frequency measures used in document searches.

In order to rank the results in a query result set, we need to assign a number to each result in the set. We start by recognizing that a query is composed of one or more terms that can be combined through Boolean operators. Each term contains exactly one keyword, which is associated to a structural element of an analogy. Example of terms are: "Concept:nucleus" and "Relation:orbits." The former is called a *concept term* and the latter a *relation term*.

We assign a weight, $w(\mu, \tau)$, to map of type μ ($\mu \in \{\text{ConceptSetMap, PrimaryRelationStructureMap, ConceptToRelationStructureMap, RelationToConceptStructureMap, RelationToRelationStructureMap}\}$) and term of type τ ($\tau \in \{\text{concept, relation}\}$) according to the number of associations in which a term of type τ is involved in a map of type μ , as shown in Table 1.

For example, a concept keyword can occur within various map elements in an analogy expression. When it occurs within a single ConceptSetMap element, there is little additional information provided about the concept beyond the source concept to which it is mapped (there is only one association with the concepts being mapped; thus a weight of 1). When it occurs within a PrimaryRelationStructureMap, however, we have more

information about the role of that concept within the analogy (the additional association occurs through the relation; thus a weight of 2). Concept keywords occurring within *ConceptToRelationStructureMaps* or *RelationToConceptStructureMaps* provide still more information about the concept's role (two more associations are added due to the internal relation inside the primary relation structure involved; thus the weight is equal to 4). Generally, such maps indicate higher order relations within the analogy, and therefore provide critical information about the basic relational structures being mapped between the target and source. Finally, if a concept keyword appears in a *RelationToRelationStructureMap*, two more associations are added because of the additional relations involved. Thus, the weight is 6 in this case.

We can now associate a score, called *Structural Information Score* (SIS), to each term $\tau : q$, where q is a keyword, τ a type of term, and a an analogy expression:

$$\text{SIS}(\tau, q, a) = \sum_{m \in \mathcal{M}(a)} \rho(q, m, a) \cdot w(h(m), \tau),$$

where $\mathcal{M}(a)$ is the set of maps of analogy expression a , $\rho(q, m, a)$ is equal to 1 if keyword q appears in map m of analogy expression a and 0 otherwise, and $h(m)$ is the type of map m . Note that the weights $w(h(m), \tau)$ are obtained from Table 1. Thus, for each analogy expression in the result set of a query Q , the SIS of each term of the query is computed as above. Finally, the rank $r(Q, a)$ of analogy expression a in the results set of Q is computed as the sum of the SISs for all the terms in the query.

The result set of a query is sorted according to this rank metric, highest to lowest. Thus, expressions providing more information about the role of the relations associated with the concept in the analogy will rank higher in the sorted list, but the user must still inspect the analogy and evaluate its pertinence for the user's purposes.

We might consider other indicators of relevance or information content for an analogy expression, but the order, structure, and detail various authors include in analogy expressions may vary significantly even for the same basic analogy. For example, how would we order two analogy expressions with identical target and source analogs, but differing level of concept details such as properties or property validity? Greater detail does not necessarily imply greater information about the role of the concept in the analogy.

The system user may want to specify that some expression characteristic is pertinent to the query purpose. For example, the user may want to select expressions having a specific Domain attribute for the target or source descriptions, such as "Biology" or "Computer Science." The system must therefore permit ordering according to such user pertinence preferences, and then order the result set according to relevance measures within the pertinence categories.

6.3. Query examples

For our MARVIN archive queries we use the Lucene text search engine, which permits construction of queries, based on the content of specific elements in the XML expressions. A simple keyword match query, *Concept:nucleus*, for example, would return all expres-

sions in the archive whose Concept elements contain at least one instance of the *nucleus* keyword. A generalization search would return expressions whose Concept elements contain a generalization of *nucleus*, such as *nucleon* or *center*. Of course, we are interested in relations as well in the analogy expressions, so we can construct a compound query, for example, Concept:nucleus AND Relation:cause, which would return expressions whose Concept elements contain the keyword *nucleus* and whose Relation element contains the keyword *cause*. The query syntax defined by Lucene also permits query of element attributes. The query SourceDescription@Domain:Biology, for example, returns all analogies in the archive whose Domain attribute for the SourceDescription element contains the keyword *Biological*.

7. A prototype for the MARVIN system

The MARVIN system (see Figure 1) is designed to use a programmable proxy server that can process Web-based Analogy Expression documents as well as those stored in the MARVIN archive. The proxy server runs an XML transformation servlet that processes the XML Analogy Expression, using an XSL stylesheet; when the user selects a specific type of visualization, the appropriate stylesheet is called. Programmers can add new stylesheets to the system to produce other visualizations and transformations. The stylesheet transforms the XML document into HTML for the tabular visualization, or into SVG for the graphical visualization. An additional transformer is used to convert the SVG graphic into JPEG format for inclusion as an image in the generated HTML page.

Note that although modern browsers such as Microsoft Internet Explorer 6 and Netscape 7 are starting to include support for XSL and SVG transformations, these efforts are still incomplete. We therefore generate standard HTML and JPG output from the MARVIN stylesheets on the server, which requires no browser processing of the analogy expressions. This server-based approach reduces the demands on the client browser, and permits generation of other output formats, such as WML [50] for display on portable wireless devices such as PDAs and Web-capable cellular phones.

The MARVIN system server is implemented using the Apache Tomcat servlet container [65]. An XML transformation engine, Cocoon [2], runs as a servlet within the container and manages the selection and processing of XML documents and XSL stylesheets. When a user requests an Analogy Expression XML document, the Tomcat servlet container passes the request to the Cocoon servlet which retrieves the XML document from the archive, retrieves the appropriate stylesheet for the type of transformation selected, and transforms the XML document contents into HTML or SVG using its XSLT processor (Xalan or Saxon). The resulting HTML is returned to the user's browser through the servlet container. Figure 13 shows the sequence diagram for this process.

This implementation achieves the primary design goals as follows:

- *Representation*: the use of a general XML content model provides a means of describing the structure and components of an analogy independent of any presentation syntax.
- *Visualization*: enables developers to produce a variety of visualization types by adding other XSL stylesheets to the system that can be processed by Cocoon.

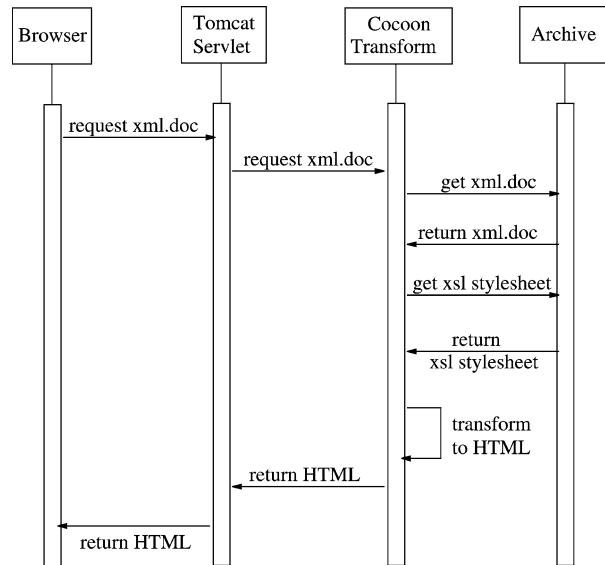


Figure 13. Sequence diagram: analogy expression transformation, XML to HTML.

- *Storage/retrieval*: enables the sharing of interesting and useful analogies with other practitioners and users by providing tools to author expressions and to add them to the MARVIN archive.

The MARVIN system stores analogy expressions in an archive that can be queried by users. The query process is implemented with the Jakarta Lucene text search engine [3], a Java servlet which includes a programmable XML document indexer and query syntax that can select for terms according to their XML element tag and content. For the MARVIN system, this permits analogy expression queries by any expression element or attribute. For example, if we wish to search the archive for analogy expressions whose concepts include *nucleus*, the query string would be ‘Concept:nucleus.’ As discussed in Section 6, Lucene permits Boolean query expressions such as ‘Concept:nucleus AND Relation:orbit.’ Figure 14 illustrates the sequence diagram for a MARVIN query. The user enters the query through the MARVIN search interface. The Tomcat and Cocoon processes pass the query to the Lucene servlet, which queries the document index (generated earlier), sorts the result set, and returns the result set to the Cocoon servlet. Cocoon then requests the search result stylesheet, transforms the list of XML documents in the result set to an HTML Web page, and passes the page back to Tomcat and then to the user’s browser.

The MARVIN user interface, shown in Figure 15, consists of a primary Web page divided into three frames. The top frame includes a link to the MARVIN Help page as well as a link to a pop-up window providing a brief general description of the system. There is also a form for the user to enter a URL for any instructional content Web page that contains links to MARVIN analogy expressions.

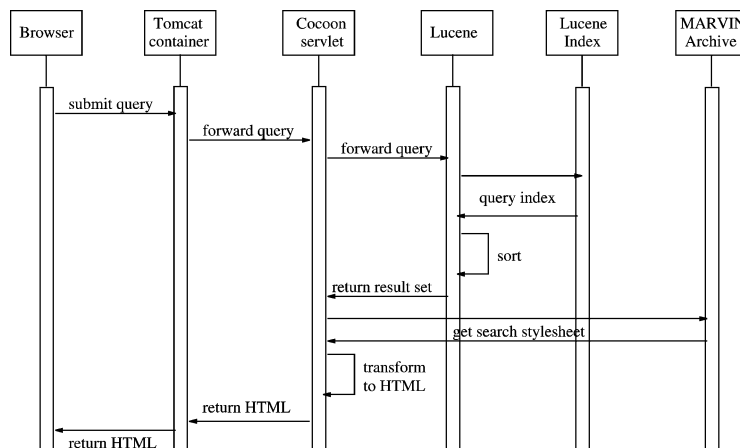


Figure 14. Sequence diagram for MARVIN archive query.

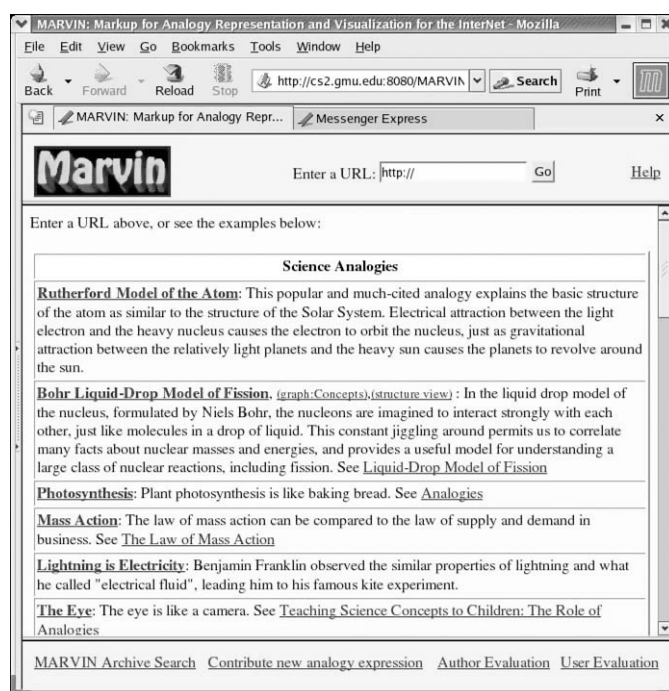


Figure 15. The MARVIN user interface.

The middle frame displays links to a collection of example analogies, used to illustrate MARVIN visualizations. Each example includes a brief description, a link to a reference for the analogy, and one or more links to visualizations produced by the MARVIN system.

The bottom frame of the main Web page includes a link to the MARVIN Archive Search, a link to a Contributor's page that allows analogy authors to upload new analogy expressions, and links to the Author Evaluation and User Evaluation surveys (discussed in Section 8).

8. Evaluation of the MARVIN system

To obtain feedback about the potential usefulness of MARVIN from analogy practitioners, we conducted a *formative evaluation* [64]. Formative evaluations can consist of one or more of the following steps: (1) expert review by practitioners, authors, or instructors; (2) one-to-one evaluation by a user with the developer; (3) small group evaluation by a group of learners or users; and field test of the system or module under realistic learning conditions. The purpose of such evaluations is to provide guidance to the developer concerning the content, functionality, and usability of a system or module under review. This provides an opportunity to adjust or to enhance the system before making final decisions about implementation. Data collected from formative evaluations can consist of developer notes from evaluator discussions and interviews, user comments, or data gathered using evaluation instruments such as questionnaires.

Additionally, because MARVIN is a Web based system with the potential to serve a large number of users and to archive a large number of analogy expressions, system sizing and retrieval performance must be considered. Section 8.2 discusses general system performance issues and presents the results of retrievals from various size archives.

8.1. Results of the formative evaluation

The formative evaluation of the MARVIN system included demonstrations and discussions with analogy practitioners and users, followed by online evaluation of the MARVIN prototype using a questionnaire for MARVIN analogy expression users and a questionnaire for MARVIN system authors. The general goals of this formative evaluation include:

- validation of the target audiences for the MARVIN system;
- validation of the usefulness of analogies in practitioner's instructional material;
- validation of the general usefulness of MARVIN analogy visualizations, and evaluation of several specific types of visualizations (tabular and graphical);
- validation of the general usefulness of archive retrieval of MARVIN analogy expressions;
- validation of the general usefulness of archive retrieval of *alternate* MARVIN analogy expressions.

To accomplish these goals, several analogy practitioners (content experts who use analogies in their work or instructional material) were identified, presented with a brief demonstration of the MARVIN system, and interviewed to obtain their comments and suggestions. Additionally, students in several instructional technology courses at GMU's

Graduate School of Education (GSE) were given an overview and demonstration of the MARVIN prototype and were asked to complete an online questionnaire about the system.

Interviews were conducted with three researchers whose work involved the use and analysis of analogies: (i) a professor of Philosophy & Religion at GMU who teaches a course on the philosophy of science, wherein the use of analogy in science discovery is discussed [55]; (ii) a professor of Education at GMU's Graduate School of Education (GSE) with research interests in problem solving and reasoning in young children (using analogies) [72]; and (iii) a professor of Education at GMU's GSE with research interests in educational technology [41].

Students in three GSE instructional technology courses (Fall'02: Instructional Strategies for the Web, Spring'03: Advanced Instructional Design, and Spring'03: Distance Learning via Networks and Telecommunications) were given brief demonstrations and presentations of the MARVIN prototype system, and were asked to volunteer to provide user or author evaluations of the system. A user evaluation questionnaire was directed at those who are presented with analogies in their instructional material. An author evaluation questionnaire was directed at those who create or select analogies for inclusion in their instructional material. Items on each questionnaire were presented using a 5-point Likert Scale response: 5: Agree Strongly; 4: Agree Somewhat; 3: Uncertain; 2: Disagree Somewhat; and 1: Disagree Strongly.

Due to space limitations, we only provide a summary of the survey results; greater detail can be found at [20]. Thirty users of analogies responded the user survey. All respondents agreed (either strongly or somewhat) that analogies helped them understand complex topics. Eighty percent of user respondents agreed that visualizations produced by the MARVIN system would help them understand analogies presented to them; when asked about the specific types of example visualizations presented by the system, 65% agreed that both the tabular and graphical visualizations would be helpful. There was strong agreement that the MARVIN archive search features were useful, with 80% agreeing about the ability to look up example analogies, and 70% agreeing about the ability to look up alternate analogies.

Six respondents identified themselves as content experts – authors of instructional content that uses analogies. All of them agreed (somewhat or strongly) that analogies were an important component of their instruction. Their responses concerning the usefulness of the MARVIN visualizations were varied. Eighty three percent agreed that MARVIN tabular visualizations help students understand analogies and 50% agreed that the graphical visualizations help their students understand analogies. All authors surveyed agreed about the usefulness of looking up example analogies and looking up alternate analogies.

8.2. *Performance and scalability*

The MARVIN system architecture is designed to use the processing power of the server, avoiding any XML transformation processing by the browser; only HTML and JPG display rendering are required at the browser. The MARVIN system server is implemented using Java servlets (Cocoon and Lucene) running within the Apache Tomcat servlet container.

Tomcat performance is sensitive to a variety of factors, notably the speed and number of processors on the server, and the version (and therefore the performance) of the JVM that runs the container. Sub-second response times at hit rates of more than 100/sec are possible even on single-CPU servers [71]. The MARVIN system does not place a high processing burden on the server/container because:

- MARVIN Analogy Expression documents are small (generally, a few KB).
- When the Cocoon servlet is first started by the Tomcat container, the servlet parameters are read from the sitemap and the main servlet is compiled *once* by the JVM; subsequent calls to the servlet do not require recompilation.
- Servlet classes are compiled once as needed when they are requested. In the case of MARVIN requests, the classes that translate XML documents to HTML tables and to SVG graphics are compiled once, and reused on subsequent calls to translate different documents.
- Cocoon caches compiled servlet components such as sitemaps and recompiles them only when generating content from updated sitemaps or stylesheets; changes to the site data do not require servlet recompilation. MARVIN files are static, so once accessed, there is no need for the servlet class that generates a visualization to be recompiled.
- There are multiple caching mechanisms in place between the user's browser and the MARVIN system – the browser cache, the user's proxy cache (if configured), the Tomcat container cache, and the Cocoon cache. Because a transformed analogy expression (that is, an HTML or SVG/JPG visualization) does not change unless the XML expression itself is changed, the browser and intermediate systems are essentially dealing with static, cacheable content. That is, once an analogy expression is transformed, no further processing is required.
- The Lucene servlet used in the MARVIN system is a high-performance text search engine designed for large document collections. There are two components of Lucene performance to consider – index creation, and searching. In the MARVIN system, index creation is initiated manually only when a new set of expressions has been added to the archive, and thus does not affect the interactive performance of the system. Lucene is capable of indexing 100 documents per second [62] or more; once indexed, query performance has been reported to be fast, according to informal Lucene e-mail user archives, and sub-second query times for large XML archives (larger than 100 MB) have been reported [39]. We ran experiments, described in the next section, to verify this assertion.

8.2.1. MARVIN archive performance As the surveys indicate, the MARVIN analogy lookup feature is a key component of the system, and therefore must be scalable in order to eventually accommodate a large number of expressions. The Jakarta Lucene text search engine used in MARVIN is an ongoing open source project, with contributions and development managed under the Apache Jakarta Project [3]. Little has been published concerning Lucene performance, other than user-contributed benchmarks [4], a student project paper [62], and a commercially sponsored whitepaper [39]. To characterize the scalability of the archive lookup component of the MARVIN system, the Lucene engine was tested using various size archives of MARVIN analogy expressions.

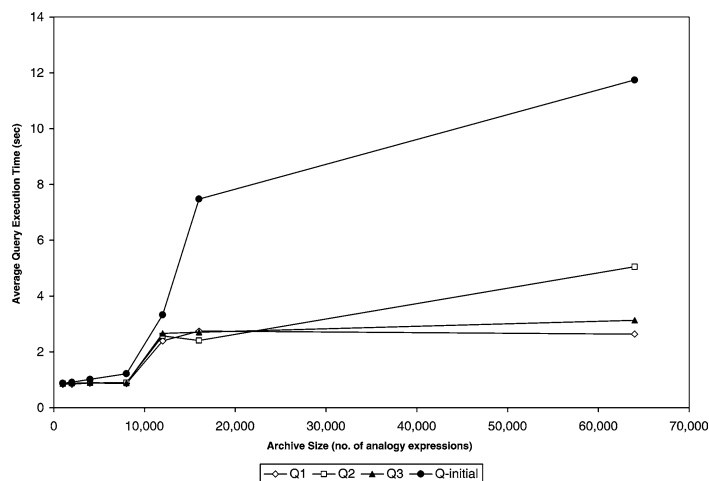


Figure 16. Average query execution time (sec) vs. size of the analogy archive (in number of analogy expressions).

To conduct the test, a procedure was needed to generate sample MARVIN analogy expressions. For this purpose, the XML Generator [78] was used. This tool can create test cases for XML applications, generating valid XML documents with random content from an XML DTD. The following steps were repeated ten times for our test:

- Create archives of 1000, 2000, 4000, 8000, 16000, and 64000 random analogy expressions using XML Generator, each conforming to the MARVIN Analogy Expression DTD, but containing random character string content within each element.
- Copy additional sample MARVIN analogy expressions to each archive.
- Create a Lucene index of each archive.
- Perform three queries for random content against the index and record the time required to complete each query:
 - Query Q1: a simple query (Concept:random-string);
 - Query Q2: a Boolean OR query (Concept:random-string Relation:random-string);
 - Query Q3: a Boolean AND query (Concept:random-string AND Relation:random-string).

Figure 16 summarizes the results of the query tests. The results show an increase in the query execution time as the size of the archive increases. The initial query for each test had the greatest execution times, but subsequent queries took substantially less time due to file system caching. Queries against a 64000 file archive were generally completed in less than 5 seconds (except for the initial query of each test run). Lucene is extremely efficient, and MARVIN analogy expression files are relatively small (averaging 10 KB each for our examples). We therefore expect that as real analogy expressions are contributed by practitioners to the MARVIN archive, query performance will be acceptable as the archive grows. These tests were conducted using a Sun Microsystems Ultra 60 server, with

dual 450 MHz processors, 512 MB memory, and NFS mounted file systems, running the Solaris 9 Operating Environment and version 1.4.1 of the Java JVM.

9. Concluding remarks

In collecting example analogies for this research, we observed an amazing range of human perception and thinking. And what people call analogies ranges from simple similes and metaphors to legitimate, complex, instructional analogies, from the humorous to the truly useful. A good analogy can generate a cascade of new ideas and hypotheses about its target, not simply for explanatory purposes but for developmental and problem solving purposes as well. For example, comparing a desired form of computer system management to the human autonomic nervous system [61] can guide engineers in creating subsystems that mimic self-managing biological organs and processes. Abstraction of the basic concepts and relations in an analogy can lead to great discoveries, such as the extension of the eye/camera analogy to the idea of a gravitational lens bending and focusing light from distant galaxies [38].

Our goal was not to simply review and discuss such analogies, however, but to capture them in some useful form, to create methods for expressing them, visualizing their structure as an aid for remembering and learning, and finding them when we need them. We started with the problem of representing analogies. Our goal was not complex completeness, for such a representation of the full human mental process and perception is probably not possible. Rather, we wanted to represent the essence of the analogies, to create a compact, human-readable representation.

The MARVIN prototype provides a minimal user interface sufficient to demonstrate the basic ideas of the system – representation, visualization, and retrieval. Comments from the evaluators indicated that further development of the interface is needed to improve the ease and understanding of querying the analogy archive.

Finally, the full range of human conceived analogies encompasses more than the types we have characterized here. Geometric, process, and even auditory or linguistic analogies might be able to be represented using XML content models. Representing them in that way would permit the application of the techniques explored in the MARVIN system to a wider range of human conceived analogies, benefiting the growing Web community of analogy practitioners and researchers in computer science, cognitive science, education, and general practice.

Appendix. Analogy Expression DTD

```
<!-- AnalogyExpression.dtd, (c) 2002 Harry J. Foxwell and Daniel A. Menasce -->
<!-- http://cs.gmu.edu/~hfoxwell/MARVIN/Docs/AnalogyExpression.dtd -->
```

```
<!ELEMENT Analogy (TargetDescription, SourceDescription, Map*)>
<!ATTLIST Analogy AnalogyName CDATA #IMPLIED Reference CDATA #IMPLIED>
```

```
<!(ELEMENT Map (ConceptSetMap | PrimaryRelationStructureMap | ConceptToRelationStructureMap |
                RelationToConceptStructureMap | RelationToRelationStructureMap)>
```

```
<!(ELEMENT ConceptSetMap (TargetConceptSet, SourceConceptSet)>
```

```
<!(ATTLIST ConceptSetMap MapLabel CDATA #IMPLIED >
```

```
<!(ELEMENT PrimaryRelationStructureMap (TargetPrimaryRelationStructure,
                                         SourcePrimaryRelationStructure)>
```

```
<!(ATTLIST PrimaryRelationStructureMap MapLabel CDATA #IMPLIED >
```

```
<!(ELEMENT ConceptToRelationStructureMap (TargetConceptToRelationStructure,
                                           SourceConceptToRelationStructure)>
```

```
<!(ATTLIST ConceptToRelationStructureMap MapLabel CDATA #IMPLIED >
```

```
<!(ELEMENT RelationToConceptStructureMap (TargetRelationToConceptStructure,
                                           SourceRelationToConceptStructure)>
```

```
<!(ATTLIST RelationToConceptStructureMap MapLabel CDATA #IMPLIED >
```

```
<!(ELEMENT RelationToRelationStructureMap (TargetRelationToRelationStructure,
                                           SourceRelationToRelationStructure)>
```

```
<!(ATTLIST RelationToRelationStructureMap MapLabel CDATA #IMPLIED >
```

```
<!(ELEMENT TargetConceptSet (ConceptDescription+)>
```

```
<!(ELEMENT TargetPrimaryRelationStructure (ConceptSet, Relation*, ConceptSet)>
```

```
<!(ELEMENT TargetConceptToRelationStructure (ConceptSet, Relation*, PrimaryRelationStructure)>
```

```
<!(ELEMENT TargetRelationToConceptStructure (PrimaryRelationStructure, Relation*, ConceptSet)>
```

```
<!(ELEMENT TargetRelationToRelationStructure (PrimaryRelationStructure, Relation*,
                                              PrimaryRelationStructure)>
```

```
<!(ELEMENT SourceConceptSet (ConceptDescription+)>
```

```
<!(ELEMENT SourcePrimaryRelationStructure (ConceptSet, Relation*, ConceptSet)>
```

```
<!(ELEMENT SourceConceptToRelationStructure (ConceptSet, Relation*, PrimaryRelationStructure)>
```

```
<!(ELEMENT SourceRelationToConceptStructure (PrimaryRelationStructure, Relation*, ConceptSet)>
```

```
<!(ELEMENT SourceRelationToRelationStructure (PrimaryRelationStructure, Relation*,
                                              PrimaryRelationStructure)>
```

```
<!(ELEMENT PrimaryRelationStructure (ConceptSet, Relation*, ConceptSet)>
```

```
<!(ELEMENT ConceptSet (ConceptDescription+)>
```

```
<!(ELEMENT TargetDescription (#PCDATA)>
```

```
<!(ATTLIST TargetDescription Domain CDATA #IMPLIED >
```

```
<!(ELEMENT SourceDescription (#PCDATA)>
```

```
<!(ATTLIST SourceDescription Domain CDATA #IMPLIED >
```

```
<!(ELEMENT ConceptDescription (Concept, Property*)>
```

```
<!(ELEMENT Concept (#PCDATA)>
```

```
<!(ELEMENT Property (#PCDATA)>
```


<!ATTLIST Property Validity (high|low|none) #IMPLIED >
 <!ELEMENT Relation (#PCDATA)>

References

- [1] Adobe SVG Viewer 3.0, www.adobe.com/svg/
- [2] Apache Cocoon Project, <http://cocoon.apache.org/>
- [3] Apache Jakarta Project, <http://jakarta.apache.org/lucene/docs/>
- [4] Apache Jakarta Project, "Lucene performance benchmarks," <http://jakarta.apache.org/lucene/docs/benchmarks.html>
- [5] Apache Software Foundation, "Xalan-Java Version 2," <http://xml.apache.org/xalan-j/>
- [6] W. B. Arthur, "Is the information revolution dead?," *Business 2.0*, March 2002, www.santafe.edu/arthur/recentpapers.html
- [7] N. Bohr, "Nobel Lecture," December 11, 1922, www.nobel.se/physics/laureates/1922/bohr-lecture.html
- [8] J. C. Bruner, *Actual Minds, Possible Worlds*, Cambridge, MA, Harvard University Press, 1986.
- [9] R. Catrambone and K. Holyoak, "Overcoming contextual limitations on problem-solving transfer," *Journal of Experimental Psychology: Learning, Memory, and Cognition* 15, 1989, 1147–1156.
- [10] A. Collins and M. Burstein, "A framework for a theory of comparison and mapping," in *Similarity and Analogical Reasoning*, eds. S. Vosniadou and A. Ortony, Cambridge, UK, Cambridge University Press, 1989.
- [11] R. Cover, "Cover pages," 2002, <http://xml.coverpages.org>
- [12] M. Cox, "Paradigm shifts and 9/11: International relations after the twin towers," *Security Dialog* 33(2), 2002.
- [13] D. Craig, N. Nersessian, and R. Catrambone, "The role of diagrammatic affordances in analogy," in *Proc. 24th Annual Conf. Cognitive Science Society*, Fairfax, VA, 2002.
- [14] J. Delahousse, "Index and knowledge drawing: a natural bridge from Topic Maps to XML SVG," 2001, www.idealliance.org/papers/xml2001papers/tm/web/04-04-02/xml2001orlandopaper.htm
- [15] epcEdit, March 2002, www.epcedit.com
- [16] B. Falkenhainer, "Learning from physical analogies: A study in analogy and the explanation process," Ph.D. Thesis, University of Illinois, Department of Computer Science, Urbana, IL, 1989.
- [17] R. W. Ferguson and K. D. Forbus, "A cognitive approach to sketch understanding," in *AAAI Spring Symposium on Sketch Understanding*, 2002.
- [18] K. Forbus, "Exploring analogy in the large," in *The Analogical Mind: Perspectives from Cognitive Science*, Cambridge, MA, MIT Press, 2001.
- [19] K. Forbus, D. Gentner, and K. Law, "MAC/FAC: A model of similarity based retrieval," *Cognitive Science* 19(2), 1995, 141–205.
- [20] H. J. Foxwell, "A Web-based system for representing, retrieving, and visualizing analogies," Ph.D. Dissertation, George Mason University, May 2003, http://cs.gmu.edu/~hfoxwell/MARVIN/Papers/HFoxwell_Dissertation.pdf
- [21] H. J. Foxwell, "Web-based representation, retrieval, and visualization of analogies," <http://cs2.gmu.edu:8080/MARVIN>
- [22] R. French, *The Subtlety of Sameness: A Theory and Computer Model of Analogy-Making*, Cambridge, MA, MIT Press, 1995.
- [23] G. Galileo, *Dialogues Concerning the Two Chief World Systems*, 1632, trans. S. Drake, 2nd Edition, University of California Press, 1967.
- [24] L. M. Garshol and G. Moore, "The XML Topic Maps (XTM) syntax," 2002, <http://xml.coverpages.org/XTM-SC34-N0328.html>
- [25] D. Gentner, "Structure-mapping: A theoretical framework for analogy," *Cognitive Science* 7, 1983, 155–170.

- [26] D. Gentner, "The mechanisms of analogical learning," in *Similarity and Analogical Reasoning*, eds. S. Vosniadou and A. Ortony, New York, Cambridge University Press, 1989.
- [27] D. Gentner and M. Jeziorski, "The shift from metaphor to analogy in Western science," in *Metaphor and Thought*, ed. A. Ortony, Cambridge, UK, Cambridge University Press, 1993.
- [28] D. Gentner and A. Stevens (eds.), *Mental Models*, London, Lawrence Erlbaum Associates, 1983.
- [29] S. Glynn, "Explaining science concepts: A teaching with analogies model," in *The Psychology of Learning Science*, eds. S. M. Glynn, R. H. Yeany, and B. K. Britton, Hillsdale, NJ, Lawrence Erlbaum Associates, 1991, pp. 219–239.
- [30] S. M. Glynn, "Teaching science with analogies: A resource for teachers and textbook authors," 1997, http://curry.edschool.virginia.edu/go/clic/nrrc/scin_ir7.html
- [31] Google, "Google technology," March 2002, www.google.com/technology/index.html
- [32] T. Grandin, "My mind is a Web browser: How people with autism think," *Cerebrum* 2(1), 2000, New York, The Charles A. Dana Foundation, www.grandin.com/inc/mind.web.browser.html
- [33] J. Heflin, J. Hendler, and S. Luke, "SHOE: A knowledge representation language for Internet applications," Technical Report CS-TR-4078 (UMIACS TR-99-71), 1999, www.cs.umd.edu/projects/plus/SHOE/pubs/#tr99
- [34] M. Hegarty, "Mental visualizations and external visualizations," in *Proc. 24th Annual Conf. Cognitive Science Society*, Fairfax, VA, 2002.
- [35] D. Hofstadter, *Fluid Concepts and Creative Analogies*, New York, Basic Books, 1995.
- [36] D. Hofstadter, "Epilog: Analogy as the core of cognition," in *The Analogical Mind: Perspectives from Cognitive Science*, Cambridge, MA, MIT Press, 2001.
- [37] K. Holyoak, D. Gentner, and B. Kokinof, "Introduction: The place of analogy in cognition," in *The Analogical Mind: Perspectives from Cognitive Science*, eds. D. Gentner, K. Holyoak, and B. Kokinov, Cambridge MA, MIT Press, 2001.
- [38] HubbleSite, "Natural lenses in space stretch Hubble's view of the Universe," 1999, <http://hubblesite.org/newscenter/archive/1999/18/text>
- [39] B. Jockman, W. Kimber, and J. Reynolds, "Case study: Enabling low-cost XML-aware searching capable of complex querying," Isogen, International, March 2002, www.isogen.com/papers/xml-aware_searching.html
- [40] M. T. Keane and F. Costello, "Setting limits on analogy: Why conceptual combination is not structural alignment," in *Analogy: A Cognitive Science Perspective*, eds. K. J. Holyoak, D. Gentner, and B. Kokinov, Cambridge, MA, MIT Press, 2001.
- [41] A. Kelly, Personal communication, September 16, 2002.
- [42] P. Klein and C. Milligan, "The role of analogies in writing to learn," in *International Conference on Ontological, Epistemological, Linguistic and Pedagogical Considerations of Language and Science Literacy*, Victoria, British Columbia, Canada, 2002, www.educ.uvic.ca/faculty/lyore/sciencelanguage/
- [43] J. Marshall, "Metacat: A self-watching cognitive architecture for analogy-making and high-level perception," Ph.D. Dissertation, Indiana University, Bloomington, IN, 1999.
- [44] P. Martin and P. Ecklund, "WebKB," 1999, <http://meganesia.int.gu.edu.au/~phmartin/WebKB/>
- [45] J. Matocha, T. Camp, and R. Hooper, "Extended analogy: An alternative lecture method," in *Proc. 29th SIGCSE Technical Symposium on Computer Science Education*, March 1998, pp. 262–266.
- [46] D. Mertz, "XML matters #7: Comparing W3C XML schemas and document type definitions," March 2001, <http://www.ibm.com/developerworks/library/x-matters7.html>
- [47] G. Miller, "WordNet: A lexical database for the English language," Princeton University, 2001, www.cogsci.princeton.edu/~wn/
- [48] Morphon, March 2002, www.morphon.com
- [49] G. Murphy, *The Big Book of Concepts*, Cambridge, MA, MIT Press, 2002.
- [50] Open Mobile Alliance, "Wireless Markup Language, version 2.0," www.openmobilealliance.org/wapdocs/wap-238-wml-20010911-a.pdf

- [51] N. Paris, "Elaborate analogies in science text: Tools for enhancing preservice teachers' knowledge and attitudes," Doctoral Dissertation, University of Georgia, 2000.
- [52] J. Park and S. Hunting (eds.), *XML Topic Maps, Creating and Using Topic Maps for the Web*, Boston, MA, Addison-Wesley, 2003.
- [53] R. Ramakrishnan and J. Gehrke, *Database Management Systems*, 3rd Edition, New York, McGraw-Hill, 2003.
- [54] M. Roblyer and J. Edwards, *Integrating Educational Technology into Teaching*, Upper Saddle River, NJ, Merrill, 2000.
- [55] D. Rothbart, Personal communication, September 16, 2002.
- [56] H. Ruan, "Analogy expression editor," 2002, <http://cs.gmu.edu/~hfoxwell/MARVIN/AnalogyEditor>
- [57] T. Ruhl, "The Altoona list of medical analogies," 2002, www.altoonafp.org/analogies.htm
- [58] T. Ruhl, "Use analogies to help patients understand medical concepts," www.aafp.org/fpr/990700fpr/18.html
- [59] G. Salton and M. McGill, *Introduction to Modern Information Retrieval*, New York, McGraw-Hill, 1983.
- [60] R. Schank, "Engines for education," Institute for the Learning Sciences, Northwestern University, November 2000, www.engines4ed.org/hyperbook
- [61] J. Shaw, "The autonomic enterprise," 2003, www.e-com.com/autonomic.html
- [62] D. Su, "Performance analysis and optimization on Lucene," 2002, www.stanford.edu/class/cs276a/projects/reports/dsu800.pdf
- [63] Sun, "J2SE 1.4," 2002, <http://java.sun.com/j2se/>
- [64] M. Tessmer, *Planning and Conducting Formative Evaluations*, London, UK, Kogan Page, 1993.
- [65] The Jakarta Project, <http://jakarta.apache.org/tomcat/index.html>
- [66] B. Tversky, J. Heiser, P. Lee, and J. Zacks, "Diagrams to augment cognition," in *Proc. 24th Annual Conf. of the Cognitive Science Society*, Fairfax, VA, 2002.
- [67] J. von Neumann, *The Computer and the Brain*, New Haven, CT, Yale University Press, 1958.
- [68] S. Vosniadou and A. Ortony (eds.), *Similarity and Analogical Reasoning*, Cambridge, Cambridge University Press, 1989.
- [69] W3C, "Recommendation 24 December 1999," 1999, www.w3.org/TR/html401/
- [70] W3C, "Scalable Vector Graphics (SVG) 1.0 specification," W3C Recommendation 04, September 2001, www.w3.org/TR/SVG
- [71] WebPerformance, 2002, <http://webperformanceinc.com>
- [72] C. White, Personal communication, September 17, 2002.
- [73] W. H. Wilson, G. S. Halford, B. Gray, and S. A. Phillips, "The STAR-2 model for mapping hierarchically structured analogs," in *The Analogical Mind: Perspectives from Cognitive Science*, eds. D. Gentner, K. Holyoak, and B. Kokinov, Cambridge, MA, MIT Press, 2001, pp. 125–159.
- [74] World Wide Web Consortium (W3C), "Extensible Markup Language (XML)," www.w3.org/XML/
- [75] World Wide Web Consortium (W3C), "Semantic Web," February 2002, www.w3.org/2001/sw
- [76] World Wide Web Consortium (W3C), "XML Schema," 2001, www.w3.org/XML/Schema
- [77] World Wide Web Consortium (W3C), "XSL Transformations (XSLT) version 1.0," W3C Recommendation 16, November 1999, www.w3.org/TR/xslt
- [78] XMLgenerator, March 2002, www.alphaworks.ibm.com/tech/xmlgenerator
- [79] XML-Spy, March 2002, www.xmlspy.com/