THE UNIVERSITY OF ALBERTA

VIEWING ANALOGICAL INFERENCE AS HYPOTHETICAL REASONING

BY

BONITA T. WONG

A THESIS

SUBMITTED TO THE FACULTY OF GRADUATE STUDIES AND RESEARCH IN PARTIAL FULFILMENT OF THE REQUIREMENTS FOR THE DEGREE OF MASTER OF SCIENCE

DEPARTMENT OF COMPUTING SCIENCE

EDMONTON, ALBERTA
SPRING 1990

ISBN 0-315-60273-2

To my parents

Jeffrey and Eva Wong

ABSTRACT

Analogical inference is the type of reasoning where inferences are made about one individual based on comparisons to another individual. Our work aims to express analogical inference as a type of reasoning called hypothetical reasoning. Hypothetical reasoning is a kind of justified assumption-based reasoning in which knowledge in the form of facts and hypotheses are used to form theories to explain why a goal inference might be true. The theories are justified according to various criteria. The justified assumption-based approach of hypothetical reasoning would seem very appropriate to capture the plausibility-based nature of analogical inference.

In attempting to formally characterize analogical inference in the paradigm of hypothetical reasoning, we identify two key aspects. These aspects are: property ascription and partial equality. Property ascription describes the assigning of a property to the target individual based on the source individual having a property. A partial equality is a relation between the target and source which justifies an ascription. A partial equality is based on equalities between properties of the target and properties of the source.

In order to demonstrate our view of analogical inference, we give a definition for it, and implement examples of it, in the THEORIST framework. We suggest an appropriate representation for domain knowledge and properties, define hypotheses to capture the key aspects, and express various criteria for justification of the hypotheses. Expression of these criteria is explored in both object level and metalevel approaches. Various domains are used for the example problems, including stories and physical systems.

ACKNOWLEDGEMENT

Many thanks to my supervisor Dr. Randy Goebel. My work has benefitted greatly from his ability to shape and communicate a cohesive vision for the Theorist framework. My work has also been sharpened through his insights into hypothetical and analogical reasoning. His sense of which ideas needed improvement was crucial, as was his knowledge about how to do research in general. As well, his great encouragement and patience have been deeply appreciated.

Many thanks to Scott Goodwin of the University of Alberta. His many insights into hypothetical reasoning have been invaluable to this work, as has his ability to contribute good criticisms on a wide variety of issues. As well, I have been one of the beneficiaries of his knowledge about hypothetical reasoning and his willingness to help others to gain in their knowledge. For his support and friendship I can only express deep thanks.

Thanks to my committee members: Dr.'s Renee Elio, Wayne Davis, and Alinda Friedman for valuable comments and criticisms. Thanks especially to Dr. Renee Elio for early ideas about analogy. As well, this work would not have come to fruition without her support and encouragement.

Thanks to the Hypothetical Reasoning Group of the University of Alberta, including George Ferguson, to whom thanks go to for his work on the Theorist interpreter, and Abdul Sattar and Patrick Fitzsimmons.

Thanks also to Russ Greiner, Fahiem Bacchus, and Larry Watanabe for their encouragements for my work and their sharing of their research wisdom with me.

Many thanks to the fellow researchers which have helped make my masters degree such an enriching and enjoyable experience, too many to name but especially: Peter Leung, Brant Coghlan, Raymond Cheng, and Gordon Fong. Thanks also to the many friends who have given me their support, prayers, and friendship; it is impossible to name all those to whom I have debts in this. I would like only to mention thanks to Enid Ma, Henry Ning, and Lingyan Shu for their help in proofreading and producing the thesis document.

Deepest thanks to my family: to my father for being an example in scholarship and scientific endeavor, and to both him and my mother, and to my brothers, Bert and Franklin, for their caring support in all areas.

Finally, I give thanks for the following inspiration and purpose: "And whatever you do, whether in word or deed, do it all in the name of the Lord Jesus, giving thanks to God the Father through him", Colossians 3:17.

TABLE OF CONTENTS

CHAPTER **PAGE**

LIST OF FIGURES

Chapter 1

Introduction

1.1. Analogical Reasoning

Analogical reasoning has been the subject of a great deal of interest in the past few decades, in both psychology and in artificial intelligence [Sternberg 77, Kedar-Cabelli 85a, Hall 87]. This type of reasoning is important to many kinds of intelligent reasoning. The use of analogies is found in scientific reasoning, commonsense reasoning, and learning. It is used to compare cases in medicine or law, draw metaphors in poetry, and learn new solutions in problem solving.

Analogical reasoning is important to investigate not only because it is vital to many other types of reasoning, but because of its general and fundamental nature. Some researchers have even proposed that analogical processes are an underlying building block of basic intelligence [Sternberg 74]. As well, analogical reasoning is useful in providing an efficient way to solve problems or learn new concepts. Rather than resolving a new problem from fundamental principles, a reasoner can build on past expertise by hypothesizing a solution based on a previous solution. When a new concept is learned, work can be saved by using an analogy to an already known concept. Analogical reasoning is also useful for creative types of reasoning, such as composition or creating new scientific theories. The drawing of novel correspondences between two concepts can lead to new insights into the subject being investigated.

Some terminology which has become common in describing analogical reasoning is as follows [Hall 87]. An analogy is a relationship of one individual, a *target*, to another individual, a *source*. An analogy can be viewed as being based on a set of correspondences between two domains of knowledge, a *target domain* and a *source domain*. The correspondences are drawn between relations of one domain and relations of the other domain. The set of correspondences is called a *map*, and each individual correspondence is called a *mapping*. The basis upon which particular mappings are determined is called the *justification* (or "confirmatory support" [Hall 87]) for the map. For example, the justification could consist of selected similarity or relevance relationships that hold between relations in the domains, or between individuals involved in those relations.

1.2 Analogical Inference

Analogical reasoning has different variations, depending on the type of problem it addresses. This dissertation focuses on *analogical inference*, which is the inference of a conjecture about one individual (the *target*), based on an analogy to another individual (the *source*).

Two main types of analogical inference are identified. They are *analogical inference explanation* and *analogical inference prediction*. In the explanation case, a reasoner is given a knowledge about the target and source (i.e., a target domain and a source domain), and a specific property that is proposed about the target. He is then asked "Is it reasonable to infer the given property about the target, on the basis of some analogy to the source?". A rephrasing of this question could be "If the given property were true about the target, would there be an analogy-based explanation why it would be true?".

An example of analogical inference explanation is a question arising as to whether a certain technique of blasting will be effective on an oil well. To answer this question, the oil driller could see if there is an explanation for assuming the blasting would be effective, based on an analogy between the current well and a previous well. Another example is a question posed to students about whether they think that in an atom the nucleus might attract the electron. They might be asked to base their answer on some set of correspondences (i.e., a *map*) between the atom and a solar system.

In these examples of analogical inference explanation, the inference (goal property) is specified, and the particular analogical map is not specified. That is, a map is established in order to support the goal inference. In the prediction case, a goal inference is not specified. The reasoner is given only a target and a source. The reasoner is then asked "What inferences would be reasonable to predict about the target, on the basis of some analogy to the source?". Any map could be established, and any inferences that follow from that map could be made.

An example of analogical inference prediction is when an oil driller compares a current oil well to a previous one, and tries to predict all likely properties of the current well based on correspondences (or *mappings*) to a previous well. The driller might also predict a second set of properties based on a different map between the two wells, and thus have two sets of predictions between which to prefer. Another example is when students are asked to hypothesize all the properties they can about an atom, based on the hint that there are possible mappings between it and a solar system.

The explanation and prediction cases of analogical inference are closely related. For instance, prediction might involve having several different goal inferences, making explanations for them, and then preferring between explanations to see which goal inferences are reasonable and hence predicted. It would seem that out of the explanation and prediction cases, the former is the simpler case, and hence needs to be understood first. Thus, in this dissertation, the focus will be on analogical inference explanation (referred to hereafter as simply *analogical inference*).

1.3 Analogical Inference Expressed as Hypothetical Reasoning

In this dissertation, analogical inference is investigated with a focus on its fundamental nature. The aim is to formalize a conceptual framework in which the basic processes underlying analogical reasoning can be characterized.

One of the most important aspects about analogical reasoning will be investigated: its plausibility-based nature. Analogical inference has been referred to as "plausible inference" [Greiner 85, Kedar-Cabelli 85a] and "heuristic inference" [Hall 87]. These descriptions emphasize that an analogical inference does not just follow from facts. It will be proposed that the plausibility aspect can be captured by reasoning that uses assumptions, a second type of knowledge that is distinguished from facts. As well, the assumptions used should not just be any possible ones, but should be deemed reasonable and have some justification for why they are assumed.

To capture the nature of analogical inference, it is proposed to use a particular type of justified assumption-based reasoning called *hypothetical reasoning* [Poole 86].

Hypothetical reasoning is a paradigm for reasoning where inferences are made through use of theories. Theories embody knowledge which is assumed for the purpose of making an inference. Hypothetical reasoning is based on a scientific reasoning cycle, and so, in it various stages, involves the formation, justification, application, testing, and revision of theories.

The goal, then, is to formalize an expression of analogical reasoning as hypothetical reasoning. It would also be desirable to have the expression be clear, general, and implementable. From the above, the thesis for this dissertation is as follows:

Analogical inference can be appropriately expressed as hypothetical reasoning, and such an expression can be done in a clear, general, and implementable manner.

1.4 Overview

In Chapter 2, some of the approaches which other researchers have used for analogical reasoning are examined. In Chapter 3, a framework for hypothetical reasoning called Theorist [Poole 86] is described in detail. Also discussed are some issues in the framework which must be addressed in order to express a type of reasoning as hypothetical reasoning. Following this, in Chapter 4, a representation for domain knowledge is proposed. Two key aspects of analogical inference are then identified and captured in appropriate hypotheses. A hypothetical reasoning framework for analogical inference which uses these two hypotheses is then defined. In Chapter 5, two approaches to expressing justifications are proposed, with implemented examples demonstrating the viability of each approach. Chapter 6 concludes with a discussion of the relation between this research and other research, the contributions of this work, and possibilities for future work.

Chapter 2

Approaches to Analogical Reasoning

2.1 Major Approaches

Research on analogical reasoning has been done from both psychological and artificial intelligence perspectives. The psychology work emphasizes the cognitive aspects of how people perform in analogical experiments (a survey of such work has been done by Sternberg [Sternberg77]). From the results of such experiements, models of reasoning are suggested. The following discussion of analogical reasoning will concentrate on artificial intelligence research, and will only discuss one approach from psychology, that of Gick and Holyoak [Gick 83, Holyoak 84a].

In the artificial intelligence approaches to analogy, one of the major distinctions for comparing approaches is whether they are data-directed or goal-directed. In the data-directed, or bottom-up approach, mappings are established on an individual basis, without a general idea of what the overall map will be like. In other words, there is no global form or structure to constrain the mappings. Rather, mappings are formed based on knowledge about each individual's properties, or each property's characteristics.

The work of Winston [Winston 80, Winston 82] illustrates this style of reasoning. His mapping is done on the basis of properties of "objects". For example, he matches Cinderella and Juliet based on their common properties of being female and being beautiful.

Another instance of data-directed mapping is the work which stems from Gentner's approach [Gentner 83, Falkenhainer 86]. Each property is mapped or not mapped according to its syntactic characteristics. For example, relations such as `greater_than(pressure(water,beaker),pressure(water,vial))` and

`greater_than(temperature(heat,coffee),temperature(heat,ice_cube))`

are mapped based on the order of the relations [Falkenhainer 86].

A second major approach besides data-directed mapping is goal-directed, or top-down, mapping. Knowledge such as an abstraction or a plan provides a general model for the map. Mappings are formed which fill a role in, or satisfy some aspect of, the guiding structure. For instance, to constrain mappings, Burstein uses relational structures, which are groups of related relations [Burstein 86]. Carbonell uses plans such as those used in means-ends planning [Carbonell 83]. Kedar-Cabelli uses schemas which include the purpose of an analogy, such as those used for explanation-based learning [Kedar-Cabelli 85b]. Greiner uses abstractions, which are schemas for a group of related relations [Greiner 85, Greiner 86].

In comparing data-directed mapping with goal-directed mapping, one can see that each approach has its own advantages. Goal-directed mapping may be better in terms of efficiency, since the guiding structure focuses the mapping process. As well, the group of mappings composing the overall map will more likely be related or relevant to one another, and will less likely be a haphazard assortment of mappings. On the other hand, the data-directed approach is advantageous for novel situations for which there are no available abstractions or schemas. The analogy may be creatively different or new, so that there are no structures abstracted from previous experiences suitable for it. In such a case, data-directed mapping would be appropriate.

The next sections of this chapter will describe some specific approaches to analogical reasoning in more detail. This description will not comprise a history of research in analogy, nor the whole spectrum of work that has been done. (Helpful overviews and syntheses of the diverse works in the field can be found in [Hall 87] and [Kedar-Cabelli 85a].) Rather, approaches will be described which are relevant to the work with hypothetical reasoning. One approach from psychology is described, that of Gick and

Holyoak, but the rest of the works described are from artificial intelligence, that is, the works of Winston, Gentner and Greiner.

2.2 Gick and Holyoak: Psychological Aspects of Analogical Inference

In the field of psychology, Gick and Holyoak have conducted experiments to observe how people use analogical situations to solve problems [Gick 83, Holyoak 84a]. In one of their experiments, subjects read two stories, the first with a problem and its solution, the second with only a problem. Subjects were to propose a solution to the second problem. It was then observed whether the subjects noticed that the stories were analogous, and consequently used this analogous relationship to infer a similar solution.

Gick and Holyoak propose that an "abstraction operator" acts on analogues to eliminate their differences and preserve their commonalities. In doing this, the operator constructs "schemas" which distinguish the differences from the commonalities. When subjects have a schema, it is proposed that they will more readily discern an analogy. Experiments confirmed that subjects more readily noticed an analogy when they had previously done an exercise of describing the commonalities between stories. The exercise was intended to encourage the formation of schemas.

Gick and Holyoak describe what an abstraction operator would accomplish. However, a weakness of their work is that they do not give any well-specified proposals regarding how an abstraction operator would work. Hence their work gives limited suggestions for a computational approach.

The results of these experiments are interesting because they suggest that if a schema is viewed as a type of justification for an analogy, then it could follow that having a schema would mean having a stronger justification (stronger than having no schema). Hence it would follow that an analogical relationship between the two stories is more likely to be assumed if there is a schema, as was indeed found. This view is what a hypothetical reasoning approach would suggest, i.e., that the schema is a type of knowledge which justifies the judgement that there is an analogy.

While Gick and Holyoak suggest that schemas are important, they propose that their use is optional, and that their formation might be a side-product of the formation of an analogy. Some of their experimental results indicate that some people identified an analogy without doing a schema exercise. These results are interesting to us in that they emphasize the individual differences in the criteria people use to perceive analogies. Hence, it would seem that an approach which is flexible in allowing different criteria, or justifications, is important. There seems to be no standard interpretation of what an analogy is. This result supports the view that analogy is more appropriately expressed as hypothetical knowledge than as factual knowledge.

2.3 Winston: Matching based on Similarity Criteria

The concept of a map based on matches between individuals is explored in Winston's work [Winston 80]. The representation, as well as the mapping process, is oriented around "objects" of the domain. Objects are matched in different combinations to form potential mappings. Each potential mapping is then ranked according to the similarity between objects.

Similarity is scored according to a metric based on the number of common properties (attributes) of individuals, relations between individuals, and special "constraint" relations. An example of a common property that is scored is that of being in a certain A-KIND-OF class. An example of a constraint relation is a relation which has been marked a priori as important by a tutor. Causal relations such as "cause" or "caused-by" are also considered constraint relations.

Besides the use of various similarity metrics to determine a best set of matches, Winston also discusses the inference of a target relation in answer to a query. The system looks for a subnet of relations which is related causally to a source instance of the query. Attempts are made to infer target instances of the subnet relations. If these inferences are successful, then the queried relation is inferred. That is, the relation is ascribed to the target from the source based on a similarity between the causal structure of the source and target.

The representation used is a type of semantic net. Nodes represent "parts" of a situation, while arcs represent relations between parts. Secondary, or "extending", relations can be "tied to" the primary act-specifying relation. An extending relation and the parts it connects are called a "supplementary description" to the primary relation and its nodes. The representation centers around the nodes, and hence the parts (or objects). Examples of analogies that are represented are between two stories and between electric-current and water-flow systems.

Winston's examples on the Cinderella story and Romeo and Juliet story provide domains which are used in some of the examples in Chapter 5. As well, the heuristic chosen to be implemented with those examples is a simple, individual-oriented, data-directed heuristic, similar to the style of mapping in Winston's work. The focus of this dissertation, however, is not on particular heuristics, but on their role in analogical inference. In contrast, while Winston's work suggests many heuristics for guiding and scoring matches, it does not investigate the interpretation of these heuristics in the reasoning process.

2.4 Gentner: Similarity Types and Structure-mapping

Gentner's work focuses on different types of similarity, and the kinds of mappings characteristic of each [Gentner 83]. There are five types of similarity she identifies: literal similarities, analogies, mere appearance matches, anomalies, and abstractions. These types are classified according to a taxonomy which depends on two types of properties being distinguished: *attributes* (which are unary relations) and *relations* (which are greater than unary relations). For example, collide(X,Y) is considered a relation, while large(X) is an attribute [Gentner 83]. A particular type of similarity is identified by the proportion of attribute mappings compared to the proportion of relation mappings.

For analogies, Gentner claims that most of the properties mapped are relations. The heuristic she suggests for analogy prefers mapping higher order relations which represent "systems" of relations. For example, the second-order relation cause(collide(X,Y), strike(Y,Z)) is considered to represent a "system" of two relations consisting of the collide(X,Y) relation and the strike(Y,Z) relation [Gentner 83]. The cause relation has higher "systematicity" than a first order relation such as affect(X,Y), and hence would be preferred for mapping.

The systematicity heuristic is part of a "structure-mapping" approach for analogies [Gentner 83]. Structure mapping is a style of mapping which consists of three main heuristics: do not map attributes, try to map relations, and prefer to map relations of higher systematicity.

The structure-mapping strategy has been implemented in a rule-based system called the Structure Mapping Engine (SME) [Falkenhainer 86]. The system is designed to be used for cognitive simulations. As such, it is flexible in the sense that various sets of matching rules can be used with the "engine" of the system. The matching rules are metalevel heuristics which guide the mapping. For instance, they could establish possible matches between relations which are identical, or assign scores to different matches, e.g., assign a higher score to a match of higher systematicity.

Maps produced by structure-mapping are influenced only by the syntax of the representation. Though it is acknowledged that there may be a limit to what a mapping strategy can achieve without semantic criteria, the claim is that mapping can be effective based on syntax alone [Falkenhainer 86, p. 276].

Gentner's work is like Winston's in that both use the strategy of forming matches on a local or individual basis. These matches are scored according to various criteria and then grouped into global matches. Gentner's work is unlike Winston's (and ours) in that it does not make inferences based on an analogy, but only finds an analogical map. Gentner's work is related to the work of this dissertation in that different kinds of similarity relations are based on different types of mappings. In the hypothetical reasoning framework, different types of equality relations are justified by different mapping criteria.

2.5 Greiner: Formalizing Useful and Abstraction-based Analogical Inference

Greiner makes an important contribution to analogical inference by investigating it with a formalized approach [Greiner 85, Greiner 86]. He gives a logic-based definition of analogical inference as a type of plausible inference [Greiner 85]. In plausible inference, inferences cannot be logically implied from what is known, and they are to be consistent with what is known. In other words, the inference has a logical independence with respect to the facts.

Below is Greiner's definition of plausible inference for the case where a particular relation r(X) is to be inferred. The symbol |~ is Greiner's symbol for a plausible inference operator.

<u>Greiner's Definition of Plausible Inference</u>

Given: F is the set of facts. F is taken to be a set of finite, consistent formulae.

Plausibly infer: r(X)

i.e. F |~ r(X)

Where

Inference is Unknown: F |≠ r(X)

Inference is Consistent: F |≠ ¬r(X)

In order to specify analogical inference, Greiner adds two conditions to plausible inference. The first is that more knowledge is provided besides the facts F. It is given that an individual T, which is from the target domain, is similar to an individual S, which is from the source domain. This is denoted T~S. An example of T~S is flowrate~current, or "flowrate is like current" [Greiner 85], where flowrate is in the water circuit domain, and current is from the electric circuit domain. The second condition added is that the relation inferred about the target is to be true of the source. For example, if the relation r(T) is to be inferred, then r(S) must be true. Greiner calls this the "Common Condition."

In addition to conditions for analogical inference, Greiner also proposes two heuristics for guiding the analogical inference. First, the inference is to be "useful" in that it helps solve a problem Problem_T that has been posed by a tutor. An example of a problem is "Find the flowrate", which could be represented as Problem_T = find-flowrate(pipe-b, X) [Jackson 86].

Second, the relation to be inferred is to be an "abstraction." An abstraction is a relation that corresponds to a group of facts that can be used to solve some set of problems. One example of an abstraction is the RKK abstraction, which is the generalization of Ohm's law for electric current and Kirchoff's law for water flow. Abstractions supply a predefined grouping of relations that are relevant to one another. An abstraction helps guide the mapping process in the choice of appropriate relations to map, and in the appropriate instantiation of arguments of relations.

Greiner expresses the requirement that the goal relation be an abstraction by the condition that F |= abstForm(r), where "abstForm" stands for "abstraction formula". An abstraction formula is defined to be an atomic formula whose relation is an abstraction. An abstraction formula for a relation r is true when r is an atomic formula, and r is a relation which is an abstraction [Greiner 85, p.56]. That is,

abstForm(r) iff atomicFormula(r) & abstraction(relation(r)),

where: atomicFormula	is a metalevel predicate which selects a subset of syntactically legal formulae,
relation	is a function which is both metalevel and second-order and maps formulae into relations,
abstraction	is a second-order predicate which selects a subset of the relations.

Greiner's definition for useful and abstraction-based analogical inference is as follows.

Greiner's Definition of Useful and Abstraction-based Analogical Inference

Given: F is the set of facts,
T~S,
$Problem_T$ is a problem relation in the target domain.

Analogically infer: r(T)

i.e. F U T~S |~ r(T)

Where

Unknown: F |≠ r(T)

Consistent: F |≠ ¬r(T)

Common: F |= r(S)

Useful: F U {r(T)} |= $Problem_T$

Abstraction-based: F |= abstForm(r(X)).

Note that the above definitions are Greiner's definitions for unary relations. He also proposes extended definitions for n-ary relations, though these are not shown here.

Greiner's work contributes a non-procedural view of analogical reasoning, whereas the great majority of work in the field has been from a procedural point of view. As well, the example of his logic-oriented methodology is important for the approach of this dissertation. His approach to formalizing analogical inference is a major influence on this work. His characterization of analogical inference as plausible inference is paralleled by the characterization of analogical inference as hypothetical inference. However, the conditions used to characterize analogical inference in a hypothetical reasoning framework are different from the conditions which Greiner uses in his framework. As well, the focus of Greiner's work is different in that it is concerned with suggesting particular heuristics to guide the analogical inference, such as the useful and abstraction-based heuristics.

2.6 Discussion and Motivation

The examples of work described demonstrate that there are a great variety of approaches to analogical reasoning. Many of these have focused on different ways to determine maps. Data-directed and goal-directed approaches to mapping have been two ways suggested. As well, multifarious similarity metrics and mapping criteria have been put forth. For example, relations [Gentner 83], attributes [Winston 80], causes [Winston 80], purposes [Kedar-Cabelli 85b], and roles [Burstein 86] have all been emphasized in turn by various mapping schemes. Criteria which emphasize semantic aspects have been suggested, as well as criteria which emphasize syntactic aspects. While the development and refinement of such heuristics is important, the plethora and variety of them can increase the complexity and confusion in the field of analogical research.

Rather than add to this complexity, it is desirable to have a different direction of inquiry. This direction is to "boil down" the analogical mechanism to its simplest form, in order to investigate its fundamental nature. The aim is to explicate a framework within which mapping processes, whatever they be, can operate. That is, a framework is desired which is general and flexible enough to express different heuristics, and yet represents a principled, explicit view of what role these heuristics play in the reasoning process. The goal, then, is to formalize a conceptual framework in which fundamental processes underpinning analogical reasoning can be characterized.

Such a framework could be a reasoning paradigm that is even more basic and general than analogical reasoning. If analogical reasoning is viewed as an expression of such a paradigm, then it would seem to be not the result of a specialized mechanism written especially for analogy, but rather the result of a more general process which uses knowledge specific to analogy. In addition, if the framework can characterize other inference processes (e.g., default reasoning, diagnostic reasoning, planning), it would give a basis for integrating those types of reasoning with analogical reasoning.

There has been little research which has made use of a more general paradigm to express analogical inference. Two instances that do are Carbonell's work with planning in a means-ends style [Carbonell 83], and Kolodner's work with operations on episodic memory structures [Kolodner 85]. It is the Theorist framework which has been chosen for the current research. This framework is believed to have the generality desired and yet be simple and suited to analogical inference. Theorist will be described in detail in the next chapter.

Chapter 3

Hypothetical Reasoning in Theorist

In this chapter the Theorist framework for hypothetical reasoning is discussed. First its description and definition are given. Then, concerns are discussed which are important to characterizing reasoning as a type of Theorist inference. Following this, Jackson's work on abstraction-based analogical inference in Theorist [Jackson 86] is described.

3.1 Description of the Theorist Framework

Theorist is a logic-based framework for hypothetical reasoning. This framework provides a specification for a logic programming system. Theories are constructed in order to infer a goal inference. Theory-based inferencing is called *explanation*.

Explanation has two main aspects. The first aspect is that Theorist distinguishes a second type of knowledge besides factual knowledge. This type is assumable, or hypothetical, knowledge. Whereas factual knowledge is certain and needs no justification, assumable knowledge can be refuted, and needs justification to be used. A Theorist knowledge base can contain both these types of knowledge. The hypothetical knowledge is in the form of schemas, or templates, for hypotheses known as *possible hypotheses*, instances of which can be used as hypotheses.

The second aspect of explanation is the method by which goals are inferred. Theorist collects instances of facts and possible hypotheses which can be used to logically imply a goal. The set of instances of possible hypotheses which are used constitutes a *theory*.

Instances of hypotheses need to be justified before they can be included in the current theory being constructed. The usual justification is consistency. That is, the theory must be consistent with the facts. Hence, a new hypothesis instance will not be accepted if it causes the current theory to be inconsistent. Other justifications may be required, depending on the type of problem or reasoning.

The following is a summary of the terminology.

G is the goal. It is a set of formulae to be explained. The formulae are assumed to be ground formulae.

F is the set of facts. It is the set of formulae known to be true in the world being represented. F is to be a consistent set.

Δ is the set of possible hypotheses. These are assumptions that are available for assuming.

Th is the theory used to explain G. It is a set of ground instances of elements of Δ.

Using the above terminology, the following is a definition of hypothetical reasoning in Theorist.

Given: F is a set of facts
G is a goal to be inferred
Δ is the set of possible hypotheses

Find: a theory Th, a set of instances of members of Δ, such that
$F \cup Th \models G$

Where:
Justification criteria for the theory are met.
(E.g., Theory is consistent: $F \cup Th$ is consistent.)

The current implementation of Theorist can handle full first order clausal logic, and is implemented in Prolog. Note that the possible hypotheses are assumed to be supplied to the knowledge base by the user. That is, the dynamic creation of hypotheses is not addressed, but rather the focus is on the use of hypotheses for constructing a theory.

The syntax used for facts and hypotheses in the Theorist knowledge base is:

```
fact <clause>.
hypothesis <hypothesis-name>: <clause>.
```

where <clause> is a clause as in Prolog. The first item of syntax indicates that the clause is an element of the facts F. The second indicates that for every instance of the hypothesis name, the clause is a member of the set of possible hypotheses Δ. Hypotheses are named in order that they can be referred to, and so that significant variables in the hypothesis can be indicated as arguments of the hypothesis name [Poole 86].

3.2 Viewing Reasoning through the Theorist Lens

There are certain concerns that are important for deciding how to express a type of reasoning as Theorist-style reasoning. The key issues are distinguishing what knowledge is factual and what is hypothetical, and determining what justifications and preference criteria are needed for theories. These aspects are briefly discussed, in order to provide the background and context for casting analogical inference into Theorist.

One of the main tasks lies in examining the particular type of reasoning in question, and determining what knowledge is suitably expressed as facts, and what as hypotheses. In particular, the knowledge cast into hypotheses often serves as a key to capturing the nature of the reasoning.

For example, in default reasoning, hypotheses represent defaults. Defaults consist of knowledge which can be assumed as long as it is consistent to do so, or, in other words, as long as there is no evidence to the contrary. For temporal reasoning, hypotheses can represent frame axioms, and therefore describe aspects of a situation assumed to remain the same from one time interval to the next. For instance, it could be assumed that a person is still alive in a situation that follows an action, e.g., a gun firing. In medical diagnostic reasoning, hypotheses might represent the assumption that a patient has a certain sickness, based on observed symptoms. In educational diagnostic reasoning, similar hypotheses might describe learning difficulties of students.

Below are examples of simple hypotheses for such types of reasoning:

```
For default reasoning:    hypothesis  sun_rises_on(Day).
                          hypothesis  flies(X) <-- isa_bird(X).

For temporal reasoning:   hypothesis  alive(do(Action, Situation)) <-- alive(Situation).

For medical diagnosis:    hypothesis  has_flu(X) <-- feverish(X), congested(X), has_aches(X).

For educational diagnosis:  hypothesis  diffic_with _subtraction_of_wholes(X)
                                            <-- has_certain_test_results(X).
```

As will be seen, in the work on analogical inference, one hypothesis (the property ascription hypothesis) will represent the assumption that the target has a certain property ascribed to it, based on comparison with the source, and a second hypothesis (the partial equality hypothesis) will represent the assumption that the target and source are related by a type of partial equality. Such hypotheses will be described in detail in Chapter 4.

Another important issue in hypothetical reasoning is *theory preference*. There are two, often overlapping, aspects to theory preference. The first concerns the *justification* of theories. Justification is based on criteria for judging whether a theory is acceptable, or valid (i.e., preferred to no theory at all). The second concerns the *ranking* of theories. Ranking is based on criteria which is used to judge between theories, to prefer one acceptable theory over another. In the work of this dissertation, the concern is mainly with justification, leaving ranking to future work.

Determining the type of justification for a hypothesis is an integral part of characterizing the hypothesis. When examining justifications, it is possible to distinguish between *object level* and *metalevel* justifications. For example, in the following hypothesis:

```
hypothesis  has_flu(X)  <--  feverish(X), congested(X), has_aches(X).
```

the indications of being feverish, congested, and having aches are object level justifications for assuming that the individual X has the flu. The justifications are object level in that they are expressed as object level relations. Poole [Poole 87] suggests that this type of hypothesis be called a *conjecture*, and the justifications be called evidence. In contrast, some hypotheses have no object level justifications; this type of hypothesis can be called a *default*. For example, the following hypothesis is a default in that there are no object level justifications required for assuming that the sun rises on some particular day:

```
hypothesis  sun_rises_on(Day).
```

Either conjectures or defaults might have metalevel justifications if there are certain criteria required of it which are expressed in the metalevel (e.g., procedurally). For example, both conjectures and defaults could have a consistency requirement which is expressed as a procedural check. Hence, consistency is an example of a metalevel justification. Other examples are criteria which depend on the syntax of the representation language. For instance, syntactic criteria for analogies could include the number of, or degree of, mapped relations.

Justifications can also be distinguished along the lines of whether they are *global* or *local* in nature. Global justifications are requirements for the theory as a whole, i.e., they constitute a property for the entire set of hypotheses. Consistency can be seen as an example of global justification. Local justifications are requirements specific to a particular hypothesis. (As will be discussed later, one of the key hypotheses, i.e., the partial equality hypothesis, has a local justification.)

For this dissertation, the challenge is to formulate appropriate facts, hypotheses, and justifications for analogical inference, in order to express it effectively in Theorist.

3.3 Jackson: Hypotheses as Abstractions for Analogy

The goal of Jackson's work [Jackson 86] is to specify Greiner's useful, abstraction-based analogical inference [Greiner 85] in the Theorist framework. He proposes a definition in the Theorist framework which parallels Greiner's definition.

In order to represent the problem-solving focus of the inference (Greiner's Useful Condition), the problem is expressed as the consequence of a fact. The antecedents of that fact express the conditions for solving the problem. One example of a problem from the water circuit domain is "Find the flowrate". A fact is used to indicate that one can solve for the flowrate at device1 if it is connected to device2 in parallel and if Kirchoff's First Law, Kirchoff's Second Law, and Ohm's law hold. This can be expressed by the following fact, where C is a current-like function, V a voltage-drop like function, R a resistance function, and L a load device set (e.g., resistors).

```
fact can-solve(C, Dev1) <- connected(Dev1, Dev2, parallel) &
                           kfl(C) &
                           ksl(V) &
                           ohms(C, V, R, L).
```

In Jackson's example, ksl(V) and ohms(C, V, R, L) cannot be proved from facts. However, the below facts are known, where rkk(C, V, R, L) represents the RKK abstraction, which is a generalization of Ohm's law and Kirchoff's law.

```
fact ksl(V) <- kk(C, V).
fact kk(C, V) <- rkk(C, V, R, L).
fact ohms(C, V, R, L) <- rkk(C, V, R, L).
```

Hence, if the abstraction relation rkk(C, V, R, L) could be assumed, the problem could be solved. It turns out that the relation can be assumed, since the possible hypothesis below has been supplied:

```
hypothesis rkk(C, V, R, L).
```

As can be seen from the example, an analogy is established through the assumption of an abstraction for a target domain, which is known to apply to a source domain (i.e., Greiner's Abstraction Condition). For example, if the variable C was instantiated by `current` from the source domain, and by `flowrate` from the target domain, then an analogical correspondence would be drawn between `current` and `flowrate`.

As can be seen from above, Jackson uses hypotheses to represent Greiner's abstractions. When an instance of a hypothesis is assumed, this mimics the use of an abstraction. When a possible hypothesis is instantiated by target domain individuals, this mimics the target instantiation of an abstraction.

Though Jackson gives suggestions for the implementation of his approach, he does not actually implement his specification. Besides his suggestion for expressing a useful and abstraction-based approach, Jackson also suggests heuristics for theory preference in analogical inference. These include preferences for theories which are simpler and have more predicitions, i.e., logical consequences which can be shown to be true.

Jackson's work on analogical inference is the first attempt at exploring analogical reasoning in the Theorist framework. However, the work of this dissertation has not been significantly influenced by Jackson's approach. As will be clearer later, the two approaches have major differences (the two works will be compared in Sec. 6.2).

Chapter 4

A Hypothetical Reasoning Framework For Analogical Inference

In this chapter, a hypothetical reasoning framework for analogical inference is described. Section 4.1 examines the concept of a domain of knowledge, and presents a representation for this concept. Section 4.2 describes two key hypotheses for characterizing analogical inference. Section 4.3 gives the overall definition of analogical inference in Theorist.

4.1 Knowledge Domains

4.1.1 The Domain Concept

In order to draw an analogy between a target and source, knowledge about the target is compared with knowledge about the source. Hence, a body of knowledge concerning the target is needed, as well as a body of knowledge concerning the source. That is, a *target domain* and a *source domain* are needed. Conceptually, a domain is a set of relations that is associated with a particular individual. Such relations are called *properties* of the individual in question.

For illustration, suppose the following assorted collection of sixteen relations is known. The relations involve various astronomical objects, people, and other things. The relations are not shown in any particular grouping or order (but are grouped in fours for readability).

shields_against(atmosphere, meteorites)
orbits(planet, sun).
sibling(sally, joe, bob).
votes_for(joe, mulroney).

orbits(moon, planet).
perpendicular_to(equator, rotation_axis).
favorite_activity(TVwatching, grade6student).
favorite_topic(solarsys, grade6student).

are_parts_of(sun, planet, moon, solarsys).
moves(solarsys).
humorous(joe).
generally_well_behaved(grade6student).

are_parts_of(gas_cloud, solarsys, galaxy).
employs(Univ_of_Alt, joe).
job(prime_minister, mulroney).
prepares_for(teacher, grade6student, juniorHighSchool).

Out of the above relations, certain ones may be deemed relevant to describing a solar system, for example. Such relations could be specified as part of a domain for a solar system. One possibility of such a domain consists of the following relations:

<u>DOMAIN FOR SOLSYS</u>
moves(solarsys).
orbits(planet, sun).
orbits(moon, planet).
are_parts_of(sun, planet, moon, solarsys).
are_parts_of(gas_cloud, solarsys, galaxy).

For a different individual, a different domain would be specified. For instance, a domain for a planet might be specified to consist of the following relations:

<u>DOMAIN FOR PLANET</u>
orbits(planet, sun).
orbits(moon, planet).
shields_against(atmosphere, meteorites).
perpendicular_to(equator, rotation_axis).
are_parts_of(sun, planet, moon, solarsys).

Other examples of domains that could be drawn from the assortment of known relations are:

DOMAIN FOR GALAXY
are_parts_of(gas_cloud, solarsys, galaxy).

DOMAIN FOR GRADE6STUDENT
generally_well_behaved(grade6student).
favorite_activity(TVwatching, grade6student).
favorite_topic(solarsys, grade6student).
prepares_for(teacher, grade6student, juniorHighSchool).

DOMAIN FOR JOE
sibling(sally, joe, bob).
employs(UnivAlt, joe).
votes_for(joe, mulroney).
humorous(joe).

Notice that a particular relation can be in more than one domain. For instance, the relation `sibling(sally,joe,bob)` could be in three domains, i.e., in the domains for sally, joe, and bob. The relation `are_parts_of(gas_cloud,solarsys,galaxy)` is in both the domain for solarsys and the domain for galaxy. The relation `orbits(moon,planet)` is in the domain for solarsys as well as the domain for planet.

It should be emphasized that a domain for an individual X is not merely the collection of relations that has X as an argument. Rather, a domain embodies knowledge about what relations describe characteristic properties. In other words, a domain does not contribute an indexing scheme but rather a type of knowledge that differs from the knowledge found in the relations themselves. To underscore this point, the following examples are given.

While the relation `votes_for(joe,mulroney)` has `mulroney` as an argument, it probably would not be included in a domain for `mulroney`. This is because while it might be suitable to include knowledge of who Joe voted for when describing Joe, this knowledge generally would not be in a description of Mulroney. A similar example is `favorite_topic(solarsys,grade6student)`, which would be relevant to a domain for Grade 6 students, but not for solar systems. Yet there are cases, perhaps the majority, where a relation having `solarsys` as an argument would indeed be relevant to `solarsys'` domain. It is up to whoever specifies the knowledge base to judge which relations are relevant to a domain and which are not. This judgement is what is expressed through the concept of a domain.

Besides the case where a relation has an individual X as an argument but is not suitable for its domain, there is the opposite case where a relation does not have an argument X but is suitable for X's domain. For example, `shields_against(atmosphere,meteorites)` is included in the domain for a planet, even though `planet` is not an argument of the relation. Knowledge about domains is needed so that relevant relations are not missed out.

The above examples illustrate the need for knowledge about domains, about associations between individuals and relations. One remaining issue is how to represent this knowledge.

4.1.2 Representing a Domain

In order to represent domains, a *domain relation* is introduced. A domain relation is a relation of the form dom(X, Prop), where X is an individual and Prop is a first order relation. Prop is said to be a property of X. A domain D_X for an individual X consists of the set of its associated relations $Prop_i$. That is, $D_X = \{ Prop_i \mid dom(X, Prop_i) \}$.

The domain for a solar system, following the previously given example, would be:

D_{solsys}
$= \{ Prop_i \mid dom(solsys, Prop_i) \}$
= {moves(solarsys),
orbits(planet, sun),
orbits(moon, planet),
are_parts_of(sun, planet, moon, solarsys),
are_parts_of(gas_cloud, solarsys, galaxy)}.

Using the domain relation, the following facts can now be added to the Theorist database in order to represent a domain for a solar system.

```
% DOMAIN FOR SOLSYS
fact dom(solarsys, moves(solarsys)).
fact dom(solarsys, orbits(planet, sun)).
fact dom(solarsys, orbits(moon, planet)).
fact dom(solarsys, are_parts_of(sun, planet, moon, solarsys).
fact dom(solarsys, are_parts_of(gas_cloud, solarsys, galaxy)).
```

Now that the concept of domains has been introduced, and a way to represent them, the analogical inference which acts upon such domains will be examined.

4.2 Hypotheses for Analogical Inference

4.2.1 Two Key Aspects of Analogical Inference

The key intuition behind a formulation of analogical inference is that the plausible reasoning aspect of it can be captured in two simple hypotheses.

The first hypothesis captures the aspect that the property to be inferred is "carried over", or "transferred" [Burstein 86], to the target from the source. To account for the case where the property to be inferred is not exactly the same as a property in the source, the property is said to be "ascribed" (rather than "transferred") to the target. A hypothesis that captures this ascribing of a property to the target is called a property ascription hypothesis. This hypothesis will be described in more detail in Section 4.2.2.

The second hypothesis captures the aspect that the source and target are plausibly related in a manner which justifies the ascription of a property between them. This relation between source and target could be named "ascriptively-related", or "analogous", or "similar with respect to the analogy". However, the name "partially equal" will be used to emphasize the point that the ascriptive relation is a type of equality. A hypothesis which captures the conjectured analogical relationship between source and target is called a partial equality hypothesis. This hypothesis will be discussed in more detail in Section 4.2.3.

4.2.2 The Property Ascription Hypothesis

It is desired to arrive at a suitable specification of property ascription for analogical inference. The concept of ascription starts from the idea that a property can be attributed to the target based on two aspects. The first aspect is that some relation holds between the target and source. The second aspect is that the source has some property which can be ascribed or transferred over. These two aspects can be expressed as follows:

```
dom(Target, Property) <-- dom(Target, some_reln(Source, Target))
                          & dom(Source, Some_property)
```

where "some_reln" is replaced by some relation, and "Some_property" by some property. The following are examples of instantiations of this knowledge:

```
dom(laundry, shrunk(laundry)) <-- dom(laundry, soaks_in(laundry, water))
                                  & dom(water, very_hot(water)).

dom(robert, dark_haired(robert)) <-- dom(robert, son_of(robert, bob))
                                     & dom(bob, black_haired(bob)).
```

However, neither of these two cases is an example of what is desired to be conveyed by the term ascription, i.e., analogical ascription. In order for the ascription to be of an analogical nature, the relationship between the target and source should not be just any one, such as soaking in or being the son of, but one that compares the target and source. Examples of comparative relations might be: equal, similar, analogous, inverse, mirror-image, opposite. These comparative relations can all be thought of as being equality-based. That is, they have meaning with respect to how much is equal (or what in particular is equal) between the individuals being compared.

Even a comparative dissimilarity can be measured by how few properties are equal in the two individuals being compared. For example, Gentner defines an anomalous relation to hold when there are few properties that are equal between the target and source [Gentner 83].

An important question is how different types of equality-based relations are defined and expressed. This question will be addressed in Section 4.2.3. For now, the role which equality-based relations in general play in property ascription will be examined.

Instead of the previously shown expression of ascription which required "some relation" between the target and source, there is now the following:

```
dom(Target, Property) <-- dom(Target, some_equality_based_reln(Source, Target))
                        & dom(Source, Some_property)
```

where "some_equality_based reln" is replaced by some equality based relation. Beyond this expression of ascription, however, one further refinement is still needed. While some equality-based relation is needed, there is a restriction that it cannot be the traditional equality (i.e., full equality, or identity). This is because part of the definition of analogy is that the target and source are not the same. Indeed, it can be seen that if a full equality was used for the equality-based relation, the ascription involved would be certain to the point of the inference process being deductive. This would then contradict the non-deductive nature of analogy. For example, consider the following:

```
dom(robert, black_haired(robert)) <-- dom(robert, equal(bob, robert)).
                                    & dom(bob, black_haired(bob)).
```

In this case, the conclusion that Robert has black hair is a certain one. It is based on Robert actually being identical to Bob (i.e., one individual is referred to by two different names or aliases). It is known for a fact that if Robert is equal to Bob, then whatever property Bob has, Robert has. Such a case does not constitute an analogy.

It is proposed, therefore, that property ascription depends not only on "some equality-based relation" (as was expressed earlier), but rather on some equality-based relation that is not full equality. Such a type of relation will be called a *partial equality* relation.

Property ascription is now expressed as:

```
dom(Target, Property) <-- dom(Target, some_partial_equality_reln(Source, Target))
                          & dom(Source, Some_property).
```

where "some_partial_equality_reln" is replaced by some partial equality relation. Below is an example of such an ascription. For brevity, the name of the relation will be changed from some_partial_equality_reln to equal_partially. The "equal_partially" name can be thought of as a common label for different possible partial equality relations.

```
dom(robert, black_haired(robert)) <-- dom( robert, equal_partially(bob, robert))
                                      & dom(bob, black_haired(bob)).
```

Now that a suitable specification of property ascription has been established, there remains the question whether it should be expressed as a fact or a hypothesis in Theorist. If ascription was based on full equality it would be viewed as a fact. But since it is not, it could be argued that it is best viewed as a hypothesis. For example, let us say that it is known that Robert is almost identical to Bob in that nearly all the properties of Robert and Bob are equal. However, even if it is known that the two individuals are almost identical, it still cannot be certain that, just because Bob has black hair, therefore Robert does also. It can only be assumed that Robert has black hair. Of course, the stronger the partial equality, the stronger the justification for assuming that Robert has black hair, but the conclusion is still an assumption.

In other words, since ascription is justified by a relation weaker than full equality, it is weaker than a fact. Hence, it is proposed that property ascription be expressed as a hypothesis. Finally, the following formulation of property ascription is established (where property_ascription(Target, Source) is the name given to the hypothesis):

```
hypothesis property_ascription(Target,Source):
    dom(Target, Property) <-- dom(Target, equal_partially(Source, Target)
                              & dom(Source, Some_property).
```

As can be seen, the particular property ascription depends a great deal on the particular partial equality. The next question concerns what the definition and expression of a partial equality relation depend on. This concern will be addressed in the next section.

4.2.3 The Partial Equality Hypothesis

Partial equality relations capture the concept that two individuals are not exactly identical and yet correspond in some manner. "Partial" describes the aspect that the sets of properties of two individuals are partially equal but not completely equal, as would be the case if the two individuals were equal. In other words, some of the properties of one individual are equal to properties of the other individual, but not all the properties are equal.

Different partial equality relations can be defined by differentiating various characteristics of the properties which are equal. Two such characteristics are the amount and type of properties. For instance, a large proportion of properties being equal might distinguish a different partial equality relation as opposed to a small proportion being equal (e.g., "similar" as opposed to "dissimilar"). The exact definition for these relations would depend on further criteria for distinguishing large and small proportions.

For another example, types of properties, such as attribute-type or relation-type, might be used to distinguish partial equalities. Gentner [Gentner 83] uses attributes and relations to define a taxonomy of comparative relations which include: anomalous, analogous, similar, metaphorical, and literally similar. If there is only a small proportion of properties which are equal, the relation is called "anomalous". If the partial equality is characterized by mainly attribute-type properties being equal, it is called "metaphorical". If it is characterized by mainly relation-type properties being equal, it is called "analogous".

Note that names for partial equality relations such as "analogous" may sometimes be confusing since there is no standard interpretation or common consensus as to what "analogous" means. The term's meaning depends on the particular criteria being used to identify the partial equality. For example, one reasoner might identify an "analogous" relation by a particular set of criteria, while another reasoner might identify a "similar" relation with the same criteria.

On the other hand, while names such as "equal_partially_1", "equal_partially_2", etc. could be used, they are not practical for common usage. In the following work, the name "equal_partially" is used as a general name for a partial equality relation. The main concern is how a partial equality relation in general can be expressed in the hypothetical reasoning framework, whatever the name and criteria chosen for it.

Though there are many possible criteria for partial equality relations, they have in common that they can be expressed in terms of criteria for an analogical map. Examples of map criteria include having mainly attribute-type mappings, or having mainly relation-type mappings.

A map is evidence for there being a partial equality relation between the target and source. When a particular type of map is established, a particular type of partial equality relation can be assumed. For the Theorist framework, it is proposed that a map can be viewed as a justification for a partial equality hypothesis. The map in turn can be justified by map criteria. An example of such an expression is:

```
hypothesis partial_equality(Target,Source):
     dom(Target, equal_partially(Source, Target)) <-- map(Source, Target).


hypothesis indiv_map(Source,Target):
     map(Source, Target) <-- map_criteria_1(Source, Target)
                           & map_criteria_2(Source, Target)
                           & map_criteria_3(Source, Target).
```

This expression of map-based justification is in the object level. That is, the map and map criteria are expressed as object level relations. This is similar to how evidence is expressed for medical conjectures. Section 5.1 will examine the object level justification of partial equality, and give specific examples of map criteria.

Another possibility for the justification is to express it in the metalevel. In this case, the justification is expressed as part of Theorist's procedures for establishing a hypothesis. This approach is similar to how a consistency justification is expressed through a consistency check procedure. With the metalevel approach, the partial equality hypothesis would appear in the object level simply as:

```
hypothesis partial_equality(Target,Source):
        dom(Target, equal_partially(Source, Target)).
```

Section 5.2 will examine the metalevel justification of partial equality. The metalevel approach is compared with the object level approach in Section 5.2.3. Whether the criteria for a partial equality is expressed in an object level or metalevel manner, the main concept is that in the Theorist framework it can be viewed as justification for a partial equality hypothesis.

4.3 Definition of Analogical Inference in Theorist

A definition of analogical inference in Theorist is now presented. The definition is shown in Fig. 4.1. The definition uses the two key hypotheses for analogical inference which have been discussed, i.e., property ascription and partial equality. As shown in Fig. 4.1, one of the definition's conditions requires the theory to include an instance of a property ascription hypothesis. Another condition is that the theory include an instance of a partial equality hypothesis.

An additional condition for the theory is that it be consistent with the facts. In other words, only theories that have been justified by consistency are acceptable. This justification is a basic criteria for the acceptability of a theory, and has been standard for hypothetical reasoning of any kind in Theorist.

While the three conditions for the theory specify the nature of the solution, conditions are also needed to specify the nature of the problem. Two conditions characterize the relation to be analogically inferred (i.e., the goal). First, the goal is to be unknown. That is, the goal cannot follow from the facts alone. This condition captures the concept that a deductive process alone is insufficient for analogical reasoning. Second, the goal is to be consistent with the facts. This condition agrees with the intuition that whatever is to be analogically conjectured should be verified (or "evaluated" [Hall 87]) for "fitting into the picture".

Together, these two conditions of an unknown and consistent goal mean that the goal is logically independent from the facts. These two conditions are standard for non-deductive cases of hypothetical reasoning in general [Poole 86]. They are also the conditions which Greiner uses to define plausible inference [Greiner 86]. Greiner specifies analogical inference as a specific case of plausible inference by adding analogy-specific conditions (the common, abstraction, and useful-for-solving-the-problem conditions). Similarly, analogical inference is specified as a case of hypothetical reasoning by adding analogy-specific conditions (use of the property ascription and partial equality hypotheses).

Note that the Theorist definition does not include Greiner's Common Condition. The Common Condition requires that the relation to be inferred must hold true for the source. Such a condition can be overly restrictive. For instance, taking an example from Burstein [Burstein 86], one might want to infer that a variable contains a value from an analogy with a box which physically holds an object. That is, the relation `dom(variable,contains(value,variable))` is to be inferred from knowing that `dom(box,physical_holds(object,box))` is true. However, this inference would not be possible if the Common Condition was required. Rather, only an inference such as `dom(variable,physical_holds(value,variable))` would be allowed.

It is more general to have a specification which allows for non-common cases rather than restrict analogical inference to common cases. Hence, the definition given does not require the problem to satisfy the Common Condition.

Fig. 4.1 summarizes the specification of analogical inference in Theorist. The set of facts F can include facts about the source domain, target domain, and other knowledge in general. For the representation, the goal G has the form `dom(T,Prop)`, where `Prop` is some property. Hence, queries will be of the form "explain `dom(T,Prop)`".

The set of possible hypotheses Δ can include hypotheses specific to analogical inference, as well as hypotheses pertaining to domain knowledge. A possible hypothesis for property ascription is denoted as $H_{property\_ascription}$, and for partial equality as $H_{partial\_equality}$.

Given: F is a set of facts
G is a goal to be analogically inferred
Δ is the set of possible hypotheses
$H_{property\_ascription} \in \Delta$
$H_{partial\_equality} \in \Delta$

Find: a theory Th, a set of instances of members of Δ, such that
$F \cup Th \models G$

Where:
Inference is Consistent: $F \not\models \neg G$
Inference is Unknown: $F \not\models G$
Theory is Consistent: Th is consistent with F
Theory assumes Property Ascription: instance_of($H_{property\_ascription}$) $\in$ Th
Theory assumes Partial Equality: instance_of($H_{partial\_equality}$) $\in$ Th

Fig. 4.1 Definition of Analogical Inference in Theorist

Chapter 5

Justification for the Partial Equality Hypothesis

In this chapter, two approaches to expressing justification for the partial equality hypothesis are examined. Firstly, an approach is proposed where the knowledge needed for analogical inference is formulated as object level hypotheses. For this approach, examples of story domain knowledge are implemented and run queries on the existing Theorist system. Through these implemented examples, it is demonstrated that analogical inference can be done without any changes to the original Theorist procedures designed for general hypothetical reasoning.

However, since the object level approach has drawbacks in terms of efficiency and expressiveness, a second approach is proposed where the justification is expressed in the metalevel. In this approach, the knowledge specific to analogical inference is expressed procedurally, rather than declaratively. Such procedures have been written, added to Theorist, and then run with various queries and instances of domain knowledge.

5.1 Expressing Justification in the Object Level

5.1.1 Hypotheses for Individual Maps and Property Mappings

In this section, an expression of justification for partial equality in the object level is explored. All the knowledge used for this justification is in the form of object level relations on object level individuals.

A justification is desired which is based on an analogical map. Hence, a relation `map(S,T)` is introduced to represent that some map holds between the source S and the target T. An example of this relation is `map(cstory,rjstory)`, which represents a map between the Cinderella story and the Romeo and Juliet story. The arguments `T` and `S` are individual constants which are to be interpreted as individuals of the domain of discourse. The following is the partial equality hypothesis with the map condition.

```
hypothesis partial_equality(T,S):
      dom(T, equal_partially(S, T))
      ←
      map(S, T).
```

This hypothesis expresses the idea that T and S can be justifiably assumed to be partially equal if there is a map-based correspondence between them. By expressing the map as an antecedent of the hypothesis, the map is linked to the partial equality at the object level.

Now it is examined as to what conditions are needed for the relation `map(S,T)` to be true. A correspondence between individuals results from a set of correspondences between properties, that is, from a set of mappings. The relation `mapping(PropS,PropT)` is introduced to represent a mapping. An example of this relation is `mapping(loves(prince_charming,cinderella),` loves(romeo,juliet)). The arguments PropS and PropT are formally interpreted as individual functions which correspond to relations of the domain of discourse.

The following is an example of a possible hypothesis which proposes a possible "blueprint" for a map.

```
hypothesis indiv_map(S,T):
    map(S, T)
    ←
      dom(S, PropS)
    & dom(T, PropT)
    & mapping(PropS, PropT).
```

This hypothesis conjectures that there exists a map between source and target if there is a mapping between a source property and a target property. An *individual map hypothesis* is a possible hypothesis which specifies conditions for a map to hold between individuals. The above example was one of the simplest examples of an individual map hypothesis, since it required only one mapping for the map. There are many possibilities for this type of hypothesis. Other examples will be given later.

The conditions for the map relation to be true determine the justification for the equal_partially relation to be true. Different justifications identify different "types" of equality for property ascription to be based on. (By a "type" of equality, a particular partial equality relation is meant.)

It is now examined as to what conditions are needed for the relation mapping(PropS,PropT) to be true. The conditions will vary greatly, depending on the particular properties. Below is one of the simplest examples of a *property mapping hypothesis*, i.e., a possible hypothesis of conditions for a mapping to hold between properties.

```
hypothesis prop_mapping(Prop, Prop):
    mapping(Prop, Prop).
```

This hypothesis conjectures that if there are two identical properties, one from the source, and one from the target, then there is a mapping between those properties. There are many possibilities for property mapping hypotheses. Other examples will be given later.

A question might arise as to where the possible hypotheses for maps and mappings come from. Because of the current scope of Theorist, it is assumed that they are supplied by the user as part of the knowledge base, just as other possible hypotheses that are not concerned with analogical inference are given to Theorist.

5.1.2 Examples

Following the introduction of the concepts of individual map hypotheses and property mapping hypotheses, four examples of analogical inference which use such hypotheses are given. The examples use straightforward criteria for maps and mappings since the intention is to illustrate an object level approach in Theorist, rather than to explore complex criteria. All the examples have been implemented using Theorist's commands for entering facts and possible hypotheses into a knowledge base, and for posing queries on the knowledge base. The answers shown in the figures for the examples depict what actual output from Theorist would be like in response to the query posed.

5.1.2.1 Story Analogy 1

This example, and several of the following examples, involve an analogy between two stories. These story analogies are based on analogies from Winston [Winston 80]. In Fig. 5.1, there are listed some source domain knowledge and target domain knowledge which are supplied to a Theorist knowledge base. Following this knowledge is a query (analogical inference to be made) posed to Theorist. After this is knowledge related to analogical inference which is relevant to the query (other analogical inference knowledge which is in the knowledge base but which is not relevant to the query is not shown). Finally, the response of Theorist to the query is shown. The response consists of justifications, a theory, and an answer.

The source is the Cinderella story (or cstory). The target is the Romeo and Juliet story (or rjstory). Besides the source and target, there are two other individuals. These are the hero and heroine. In both stories, the hero loves the heroine, and woos the heroine. It is known that the heroine loves the hero in return in the Cinderella story, but it is not known in the Romeo and Juliet story. It is also known that the hero and heroine met at a ball in the Cinderella story, and die together in the Romeo and Juliet story.

The query is whether, in the Romeo and Juliet story, the heroine loves the hero in return.

A property ascription hypothesis is given for ascribing the relation loves_in_return on target individuals. As well, a partial equality hypothesis is given.

There are three individual map hypotheses given. The first (called indiv_map1) hypothesizes a map based on one mapping. The relation `map(S,T)` can be assumed if: the source has a property, the target also has this same property, and the relation `mapping(Prop,Prop)` is true for these properties.

To prove that an instance of `mapping(Prop,Prop)` is true, a property mapping hypothesis is needed. One is given (named prop_mapping1); it proposes that any two properties that are identical can form a mapping.

The second individual map hypothesis (named indiv_map2) requires two mappings. Two pairs of identical properties are needed. The two pairs of properties must be distinct, as is specified by the condition `distinct([Prop1,Prop2])`. Note that it is assumed that the relation `distinct(List_of_props)` has been defined elsewhere in the knowledge base. The relation is a straightforward one; it is true if the its argument, a list, has no duplicate elements (i.e., the relation is a test for uniqueness in a list).

The third individual map hypothesis (named indiv_map3) requires three mappings, involving three pairs of identical properties. Its conditions are similar to those for the other two individual map hypotheses.

Theorist's first theory includes an instance of the indiv_map1 hypothesis (see Fig. 5.1). The theory basically conjectures a map consisting of one mapping between the properties `love(hero,heroine)` and `love(hero,heroine)`. The (metalevel) justification for the theory is consistency. (The displaying of "Consistent_Theory" as the justification is for the user's benefit, and is printed after all the consistency checks on the hypotheses in the theory have been satisfied.) As is shown in the displayed answer, the queried relation is successfully explained by the theory.

Similarly, Theorist produces a second theory which includes a different instance of the same indiv_map1 hypothesis. The hypothesized map involves one mapping between `woos(hero, heroine)` and `woos(hero,heroine)`.

A third theory can also explain the queried relation. This third theory uses an instance of the indiv_map2 hypothesis, and has a map of two mappings, instead of one. The two mappings are between `love(hero,heroine)` and `love(hero,heroine)`, and between `woos(hero,heroine)` and `woos(hero,heroine)`.

Note that the third individual map hypothesis was not suitable to be used in an explanation because three pairs of identical properties could not be found in the domain knowledge.

The three responses which Theorist gives to explain the query are an example of how multiple theories can arise to explain an analogical inference. This coincides with the intuition that there can be multiple, different maps between a source and target. Note that, in the example given, there was considerable overlap between the three theories, in that the mappings in the first and second maps were also in the third map. However, there are cases where the multiple maps arising from multiple theories are completely different. In the given example, whether the first, second, or third theory is preferred is an issue of theory preference. This issue is not pursued here, but it is simply mentioned that another level of criteria, i.e. ranking criteria, can be used to rank different theories. Possibilities for future work in the area of ranking criteria will be suggested in Sec. 6.4.

STORY ANALOGY 1

Knowledge for source cstory (Cinderella Story)

```
fact dom(cstory, loves(hero, heroine)).
fact dom(cstory, woos(hero, heroine)).
fact dom(cstory, loves_in_return(heroine, hero)).
fact dom(cstory, meet_at_a_ball(hero, heroine)).
```

Knowledge for target rjstory (Romeo and Juliet Story)

```
fact dom(rjstory, loves(hero, heroine)).
fact dom(rjstory, die_together(hero, heroine)).
fact dom(rjstory, woos(hero, heroine)).
```

QUERY is: explain dom(rjstory, loves_in_return(heroine, hero)).

Knowledge for Analogical Inference

```
hypothesis property_ascription(T,S):
        dom(T, loves_in_return(At, Bt))
        ←
          dom(S, loves_in_return(As, Bs))
        & dom(T, equal_partially(S,T)).

hypothesis partial_equality(T,S):
        dom(T, equal_partially(S,T))
        ←
        map(S, T).
```

Individual Map Hypotheses

```
hypothesis indiv_map1(S,T):
        map(S, T)
        ←
          dom(S, Prop)
        & dom(T, Prop)
        & mapping(Prop, Prop).

hypothesis indiv_map2(S,T):
        map(S, T)
        ←
          dom(S, Prop1)
        & dom(T, Prop1)
        & mapping(Prop1, Prop1)
        & dom(S, Prop2)
        & dom(T, Prop2)
        & mapping(Prop2, Prop2)
        & distinct([Prop1, Prop2]).
```

```
hypothesis indiv_map3(S,T):
        map(S, T)
        <-
          dom(S, Prop1)
        & dom(T, Prop1)
        & mapping(Prop1, Prop1)
        & dom(S, Prop2)
        & dom(T, Prop2)
        & mapping(Prop2, Prop2)
        & dom(S, Prop3)
        & dom(T, Prop3)
        & mapping(Prop3, Prop3)
        & distinct([Prop1, Prop2, Prop3]).
```

Property Mapping Hypotheses

```
hypothesis prop_mapping1(Prop, Prop):
        mapping(Prop, Prop).
```

```
META JUSTIFICATION is: [Consistent_Theory].

THEORY is: [property_ascription(rjstory, cstory),
            partial_equality(rjstory, cstory),
            indiv_map1(cstory, rjstory),
            prop_mapping1(loves(hero, heroine), loves(hero, heroine))].

ANSWER is: [dom(rjstory, loves_in_return(heroine, hero))].
```

```
META JUSTIFICATION is: [Consistent_Theory].

THEORY is: [property_ascription(rjstory, cstory),
            partial_equality(rjstory, cstory),
            indiv_map1(cstory, rjstory),
            prop_mapping1(woos(hero, heroine), woos(hero, heroine))].

ANSWER is: [dom(rjstory, loves_in_return(heroine, hero))].
```

```
META JUSTIFICATION is: [Consistent_Theory].

THEORY is: [property_ascription(rjstory, cstory),
            partial_equality(rjstory, cstory),
            indiv_map2(cstory, rjstory),
            prop_mapping1(loves(hero, heroine), loves(hero, heroine)),
            prop_mapping1(woos(hero, heroine), woos(hero, heroine))].

ANSWER is: [dom(rjstory, loves_in_return(heroine, hero))].
```

Figure 5.1 Story Analogy 1

5.1.2.2 Story Analogy 2

In the first Story Analogy, all the mappings were between identical properties because the source properties were all identical to the target properties. In the second example, (Fig. 5.2), a source property is changed so that its name differs from the name of its corresponding target property. The property `woos(hero,heroine)` is changed to `pursues(hero,heroine)`.

The hypothesis indiv_map1 that was used for Story Analogy 1 is still applicable. An instance of it is included in a theory involving one mapping between `loves(hero,heroine)` and `loves(hero,heroine)`. (This theory is the same as the second theory shown in Story Analogy 1.)

The hypothesis indiv_map2 is no longer applicable. This is because it specifies two identical-property mappings, whereas only one identical-property mapping is now possible. A new individual map hypothesis is given, named indiv_map4. This hypothesis involves two mappings, as the previous indiv_map2 hypothesis did. However, it differs in its second mapping . Instead of specifying a mapping between any two identical properties, a mapping is specified between the two specific properties of `pursues(As,Bs)` and `woos(At,Bt)`.

To establish `mapping(pursues(As,Bs), woos(At,Bt))`, a new property mapping hypothesis is needed. The possible hypothesis named prop_mapping2 is given. An instance of it is true if the arguments of the pursues function are identical to the arguments of the woos function. For example, the relation `mapping(pursues(hero,heroine), woos(hero,heroine))` could be assumed via this hypothesis.

The second theory shown in Fig. 5.2 includes instances of the new indiv_map4 and prop_mapping2 hypotheses. The first mapping is between the properties `loves(hero,heroine)` and `loves(hero,heroine)`; the second mapping is between the properties `pursues(hero,heroine)` and `woos(hero,heroine)`.

STORY ANALOGY 2

Knowledge for source cstory (Cinderella Story)

```
fact dom(cstory, loves(hero, heroine)).
fact dom(cstory, pursues(hero, heroine)).
fact dom(cstory, loves_in_return(heroine, hero)).
fact dom(cstory, meet_at_a_ball(hero, heroine)).
```

Knowledge for target rjstory (Romeo and Juliet Story)

```
fact dom(rjstory, loves(hero, heroine)).
fact dom(rjstory, die_together(hero, heroine)).
fact dom(rjstory, woos(hero, heroine)).
```

QUERY is: explain dom(rjstory, loves_in_return(heroine, hero)).

Knowledge for Analogical Inference

```
hypothesis property_ascription(T,S):
    dom(T, loves_in_return(At, Bt))
    ←
     dom(S, loves_in_return(As, Bs))
    & dom(T, equal_partially(S,T)).

hypothesis partial_equality(T,S):
    dom(T, equal_partially(S,T))
    ←
    map(S, T).
```

Individual Map Hypotheses

```
hypothesis indiv_map1(S,T):
    map(S, T)
    ←
      dom(S, Prop)
    & dom(T, Prop)
    & mapping(Prop, Prop).

hypothesis indiv_map4(S,T):
    map(S, T)
    ←
      dom(S, Prop1))
    & dom(T, Prop1)
    & mapping(Prop1, Prop1)
    & dom(S, pursues(As, Bs))
    & dom(T, woos(At, Bt))
    & mapping(pursues(As, Bs), woos(At, Bt)).
```

<u>Property Mapping Hypotheses</u>

```
hypothesis  prop_mapping1(Prop, Prop):
        mapping(Prop, Prop).

hypothesis  prop_mapping2(pursues(A,B), woos(A,B)):
        mapping(pursues(A,B), woos(A,B)).
```

```
META JUSTIFICATION is: [Consistent_Theory].

THEORY is: [property_ascription(rjstory, cstory),
            partial_equality(rjstory, cstory),
            indiv_map1(cstory, rjstory),
            prop_mapping1(loves(hero, heroine), loves(hero, heroine))].

ANSWER is: [dom(rjstory, loves_in_return(heroine, hero))].
```

```
META JUSTIFICATION is: [Consistent_Theory].

THEORY is: [property_ascription(rjstory, cstory),
            partial_equality(rjstory, cstory),
            indiv_map4(cstory, rjstory),
            prop_mapping1(loves(hero, heroine), loves(hero, heroine)),
            prop_mapping2(pursues(hero, heroine), woos(hero, heroine))].

ANSWER is: [dom(rjstory, loves_in_return(heroine, hero))].
```

Figure 5.2 Story Analogy 2

5.1.2.3 Story Analogy 3

In Story Analogies 1 and 2, the target domain had individuals which had identical counterparts in the source domain. Specifically, the target had the individuals hero and heroine, and the source had the same individuals. In Story Analogy 3, (Fig. 5.3), the individuals prince_charming and romeo replace the individual hero in the source and target domains respectively.

Recall that an instance of the previous property mapping hypothesis prop_mapping1 was used for inferring `mapping(loves(hero,heroine),loves(hero, heroine))`. However, the same hypothesis does not work for the inferring of `mapping(loves(prince_charming,heroine),loves(romeo,heroine))`, since the two properties are no longer identical. A different property mapping hypothesis is needed.

One possibility is a hypothesis which allows any individual to match for the first argument of the properties. Such a hypothesis could be:

```
hypothesis prop_mapping3(loves(Any_indiv1, B), loves(Any_indiv2, B)):
        mapping(loves(Any_indiv1, B), loves(Any_indiv2, B)).
```

Such a hypothesis could be instantiated to infer that the prince loves the heroine:

```
mapping(loves(prince_charming, heroine), loves(romeo, heroine)).
```

However, it could also be instantiated to infer that the prince loves romeo's horse (or any other individual for that matter):

```
mapping(loves(prince_charming, heroine), loves(horse_of_romeo, heroine)).
```

While a correspondence between the loves of prince charming and romeo's horse might be acceptable, this example indicates it might be desirable to have a mapping hypothesis with more constraints. Specifically, a hypothesis could have the constraint that the individuals involved in the properties have a correspondence of their own. An example of such a hypothesis is:

```
hypothesis  prop_mapping3(loves(As,B), loves(At,B)):
      mapping(loves(As,B), loves(At,B))
      ←
      dom(At, equal_partially(As, At)).
```

This hypothesis requires a more stringent justification for a mapping. It defines an acceptable mapping not just based on the properties' relation names alone, but also on their arguments. An instance of this hypothesis could be:

```
hypothesis  prop_mapping3(loves(prince_charming,heroine),
                                        loves(horse_of_romeo,heroine)):
      mapping(loves(prince_charming,heroine), loves(horse_of_romeo,heroine))
      ←
      dom(horse_of_romeo, equal_partially(prince_charming, horse_of_romeo)).
```

Now, an analogous correspondence will need to be established between romeo's horse and the prince before a mapping is established.

Similarly, a new property mapping hypothesis, named prop_mapping4, which also requires a sub-correspondence between individuals, is provided for mapping the pursues and woos properties (see Fig. 5.3). The new prop_mapping4 is used for the pursues-woos mapping instead of prop_mapping2, in the same manner that the new prop_mapping3 was used for the loves-loves mapping instead of prop_mapping1.

In order to establish a partial equality between prince_charming and romeo, their properties will need to be mapped in turn. Knowledge has been provided that they are both brave, and that the prince is charming, while romeo is dashing. A new map hypothesis, indiv_map5, specifies a possible set of two mappings, which will used for a map between prince_charming and romeo.

The property mapping hypotheses new to this example embody the recursive nature of partial equalities. The partial equality between the two stories depends upon partial equalities of other individuals in their domains. With respect to the nature of analogy, this recursiveness coincides with the intuition that two individuals are analogous partly because their parts or components are analogous, and that these component individuals are in turn analogous because of their parts, etc. Hence it can be seen that there is a nesting of mappings between properties, subproperties (properties of individuals involved in the original properties), sub-subproperties, etc. This recursive approach to mapping is very similar to the approach in [Rumelhart 81] and [Moore 74] (the Merlin system).

In the examples shown, there is a simplifying assumption that there will exist an end case of matchable properties to stop the recursion. That is, there will be a pair of properties which involve individuals which are identical or which have already been mapped. Thus, the nesting will not be infinite. In more complex examples, metalevel control knowledge may be needed to specify constraints for the recursion. Such knowledge could specify the number of times, for instance, that recursion is allowed.

STORY ANALOGY 3

Knowledge for source cstory (Cinderella Story)

```
fact dom(cstory, loves(prince_charming, heroine)).
fact dom(cstory, pursues(prince_charming, heroine)).
fact dom(cstory, loves_in_return(heroine, prince_charming)).
fact dom(cstory, meet_at_a_ball(prince_charming, heroine)).

fact dom(prince_charming, brave(prince_charming)).
fact dom(prince_charming, charming(prince_charming)).
```

Knowledge for target rjstory (Romeo and Juliet Story)

```
fact dom(rjstory, loves(romeo, heroine)).
fact dom(rjstory, die_together(romeo, heroine)).
fact dom(rjstory, woos(romeo, heroine)).

fact dom(romeo, brave(romeo)).
fact dom(romeo, dashing(romeo)).
```

QUERY is: explain dom(rjstory, loves_in_return(heroine, romeo)).

Knowledge for Analogical Inference

```
hypothesis property_ascription(T,S):
        dom(T, loves_in_return(At, Bt))
        ←
         dom(S, loves_in_return(As, Bs))
        & dom(T, equal_partially(S,T)).

hypothesis partial_equality(T,S):
        dom(T, equal_partially(S,T))
        ←
        map(S, T).
```

Individual Map Hypotheses

```
hypothesis indiv_map4(S,T):
        map(S, T)
        ←
         dom(S, loves(As, Bs))
        & dom(T, loves(At, Bt))
        & mapping(loves(As, Bs), loves(At, Bt))
        & dom(S, pursues(As, Bs))
        & dom(T, woos(At, Bt))
        & mapping(pursues(As, Bs), woos(At, Bt)).
```

```
hypothesis indiv_map5(S,T):
    map(S, T)
    ←
     dom(S, charming(S))
    & dom(T, dashing(T))
    & mapping(charming(S), dashing(T))
    & dom(S, brave(S))
    & dom(T, brave(T))
    & mapping(brave(S), brave(T)).
```

<u>Property Mapping Hypotheses</u>

```
hypothesis prop_mapping1(Prop, Prop):
    mapping(Prop, Prop).


hypothesis prop_mapping3(loves(As,B), loves(At,B)):
    mapping(loves(As,B), loves(At,B))
    ←
    dom(At, equal_partially(As, At)).

hypothesis prop_mapping4(pursues(As,B), woos(At,B)):
    mapping(pursues(As,B), woos(At,B))
    ←
    dom(At, equal_partially(As, At)).

hypothesis prop_mapping5(charming(S), dashing(T)):
    mapping(charming(S), dashing(T)).
```

```
META JUSTIFICATION is: [Consistent_Theory].

THEORY is: [property_ascription(rjstory, cstory),
            partial_equality(rjstory, cstory),

            indiv_map4(cstory, rjstory),
            prop_mapping3(loves(prince_charming, heroine), loves(romeo, heroine)),
            prop_mapping4(pursues(prince_charming, heroine), woos(romeo, heroine)),

            indiv_map5(prince_charming, romeo),
            prop_mapping5(charming(prince_charming), dashing(romeo)),
            prop_mapping1(brave(prince_charming), brave(romeo)),

ANSWER is: [dom(rjstory, loves_in_return(heroine, romeo))].
```

Figure 5.3 Story Analogy 3

5.1.2.4 Story Analogy 4

The fourth Story Analogy is similar to the third, except it is of greater complexity. There are two major additions to the example. Firstly, the individual heroine is substituted for by the non-identical individuals of cinderella and juliet. With this change, an additional condition is added to the property mapping hypotheses for the loves-loves mapping and the pursues-woos mapping. These hypotheses must now establish a second nested partial equality besides that between prince charming and romeo. That is, they must also establish that cinderella and juliet have a map between them. See Fig. 5.4. The new property mapping hypotheses are those numbered 6 and 7 (instead of the previous ones numbered 3 and 4). The map between cinderella and juliet is established by the new knowledge that they both have the property of being lovely.

A second addition to the analogy is included to illustrate a deeper level of nesting in the mappings. This addition is the knowledge that prince_charming is a prince, romeo is a boy, and both a prince and a boy are males. This knowledge could be represented as the following:

```
fact dom(prince_charming, prince(prince_charming)).
fact dom(romeo, boy(romeo)).
fact dom(X, male(X)) ← dom(X, prince(X)).
fact dom(X, male(X)) ← dom(X, boy(X)).
```

In such a case, mapping would be done between the prince and boy properties, which would in turn lead to mapping between the male properties. However, in the example which has been implemented, a choice was made to represent the concepts of prince, boy, and male as individuals rather than as relations. This choice was made in order to more closely imitate Winston's examples, where the "ISA" relation is used between individuals. Hence, the knowledge is represented as the following:

```
fact dom(prince_charming, isa(prince_charming, prince)).
fact dom(romeo, isa(romeo, boy)).
fact dom(prince, isa(prince, male)).
fact dom(boy, isa(boy, male)).
```

Establishing a map between prince_charming and romeo involves nesting to a third level of individuals. The inference of a partial equality between cstory and rjstory requires a partial equality between prince_charming and romeo, which in turn requires a partial equality between prince and boy. Nesting of property mappings is done through "ISA" properties, in the style of an ISA-hierarchy representing individuals and their ancestors. One of the ISA chains is prince_charming--ISA--prince--ISA-male, while the other is romeo--ISA--boy--ISA--male.

In more complex cases, there may be a need for metalevel control knowledge to specify how far up the ISA hierarchy the mapping goes. For example, there may be an absolute number of levels allowed, or a range, or a proportion of the total length of the ISA chain. In some cases there may be more than one possible ISA chain for justifying a map between two individuals. In such cases, various heuristics might use both chains and then compare, or use only one chain chosen according to criteria such as the length of chain, or class of chain.

The new hypothesis indiv_map6 adds the ISA mapping to the map. The hypotheses indiv_map7, prop_mapping8, and prop_mapping9 establish maps through ISA chaining. Indiv_map7 establishes a map between two individuals if they both have ISA properties that are mappable. Prop_mapping8 is the end-of-the-chain case, where the two ancestors mapped are identical; prop_mapping9 is the middle-of-the-chain case, where the two ancestors are different and need a further map.

STORY ANALOGY 4

Knowledge for source cstory (Cinderella Story)

```
fact dom(cstory, loves(prince_charming, cinderella)).
fact dom(cstory, pursues(prince_charming, cinderella)).
fact dom(cstory, loves_in_return(cinderella, prince_charming)).
fact dom(cstory, meet_at_a_ball(prince_charming, cinderella)).

fact dom(prince_charming, brave(prince_charming)).
fact dom(prince_charming, charming(prince_charming)).
fact dom(prince_charming, isa(prince_charming, prince)).

fact dom(cinderella, lovely()).
```

Knowledge for target rjstory (Romeo and Juliet Story)

```
fact dom(rjstory, loves(romeo, juliet)).
fact dom(rjstory, die_together(romeo, juliet)).
fact dom(rjstory, woos(romeo, juliet)).

fact dom(romeo, brave(romeo)).
fact dom(romeo, dashing(romeo)).
fact dom(romeo, isa(romeo, boy)).

fact dom(juliet, lovely(juliet)).
```

General Knowledge

```
fact dom(prince, isa(prince, male)).
fact dom(boy, isa(boy, male)).
```

QUERY is: explain dom(rjstory, loves_in_return(juliet, romeo)).

Knowledge for Analogical Inference

```
hypothesis property_ascription(T,S):
        dom(T, loves_in_return(At, Bt))
        ←
          dom(S, loves_in_return(As, Bs))
        & dom(T, equal_partially(S, T)).

hypothesis partial_equality(T,S):
        dom(T, equal_partially(S, T))
        ←
        map(S, T).
```

<u>Individual Map Hypotheses</u>

```
hypothesis indiv_map1(S,T):
    map(S, T)
    ←
     dom(S, Prop)
    & dom(T, Prop)
    & mapping(Prop, Prop).

hypothesis indiv_map4(S,T):
    map(S, T)
    ←
      dom(S, loves(As, Bs))
    & dom(T, loves(At, Bt))
    & mapping(loves(As, Bs), loves(At, Bt))
    & dom(S, pursues(As, Bs))
    & dom(T, woos(At, Bt))
    & mapping(pursues(As, Bs), woos(At, Bt)).

hypothesis indiv_map6(S,T):
    map(S, T)
    ←
     dom(S, charming(S))
    & dom(T, dashing(T))
    & mapping(charming(S), dashing(T))
    & dom(S, brave(S))
    & dom(T, brave(T))
    & mapping(brave(S), brave(T))
    & dom(S, isa(S, As))
    & dom(T, isa(T, At))
    & mapping(isa(S, As), isa(T, At)).

hypothesis indiv_map7(S,T):
    map(S, T)
    ←
      dom(S, isa(S, Parent_of_S))
    & dom(T, isa(T, Parent_of_T))
    & mapping(isa(S, Parent_of_S), isa(T, Parent_of_T)).
```

<u>Property Mapping Hypotheses</u>

```
hypothesis prop_mapping1(Prop, Prop):
    mapping(Prop, Prop).

hypothesis prop_mapping5(charming(S), dashing(T)):
    mapping(charming(S), dashing(T)).
```

```
hypothesis prop_mapping6(loves(As,Bs), loves(At,Bt)):
        mapping(loves(As,Bs), loves(At,Bt))
        ←
        dom(At, equal_partially(As, At))
        & dom(Bt, equal_partially(Bs, Bt)).

hypothesis prop_mapping7(pursues(As,Bs), woos(At,Bt)):
        mapping(pursues(As,Bs), woos(At,Bt))
        ←
        dom(At, equal_partially(As, At))
        & dom(Bt, equal_partially(Bs, Bt)).

hypothesis prop_mapping8(isa(S, A), isa(T, A)):
        mapping(isa(S, A), isa(T, A)).

hypothesis prop_mapping9(isa(As), isa(At)):
        mapping(isa(S, As), isa(T, At))
        ←
        dom(At, equal_partially(As, At)).
```

```
META JUSTIFICATION is: [Consistent_Theory].

THEORY is: [property_ascription(rjstory, cstory),
            partial_equality(rjstory, cstory),

            indiv_map4(cstory, rjstory),
            prop_mapping6(loves(prince_charming, cinderella), loves(romeo, juliet)),
            prop_mapping7(pursues(prince_charming, cinderella), woos(romeo, juliet)),

            indiv_map6(prince_charming, romeo),
            prop_mapping5(charming(prince_charming), dashing(romeo)),
            prop_mapping1(brave(prince_charming), brave(romeo)),
            prop_mapping9(isa(prince_charming, prince), isa(romeo, boy)),

            indiv_map7(prince, boy),
            prop_mapping8(isa(prince_charming, male), isa(romeo, male)),

            indiv_map1(cinderella, juliet)
            prop_mapping1(lovely(cinderella), lovely(juliet))].

ANSWER is: [dom(rjstory, loves_in_return(juliet, romeo))].
```

Figure 5.4 Story Analogy 4

5.1.3 Discussion of the Object Level Approach

From the above examples, it can be seen how object level hypotheses that express criteria for maps and mappings can be used to justify a partial equality. Analogical inference is successfully performed without changing any of the Theorist procedures for proving facts, justifying hypotheses, or forming theories. In short, the mechanisms that Theorist has for explaining queries in general do not have to be augmented or changed in order to handle analogical inference. This result points to the suitability of viewing analogical inference as hypothetical reasoning.

Another benefit from seeing how justification can be done in the object level is that maps and mappings are expressed declaratively, and hence their nature is clear and explicit.

Whereas clarity and explictness might be the strengths of an object level approach, its weaknesses lie in efficiency and expressiveness. The object level approach involves the enumeration of a great number of hypotheses. This is mainly due to the necessary reification of first order relations of the domain into functions so that higher order relations such as dom(X, PropX) can make statements about them within first order logic. Since these first order relations (properties) have been cast as functions instead of relations, they cannot be quantified over. One result of this is the necessity to enumerate one hypothesis for each map which consists of a different number of mappings. For example, one individual map hypothesis would specify a map of one mapping, a second hypothesis a map of two mappings, etc. Such cases were seen in Story Analogy 1, in Section 5.1.2.1. A second result of relation names needing to be explicitly referenced is that a hypothesis needs to be enumerated for each specific combination of property mappings for a map. Likewise, each specific set of conditions for a property mapping also requires a particular hypothesis. Because of the large number of hypotheses involved, there is greater complexity and less efficiency in the analogical inference.

Besides being weak in efficiency, the object level approach is limited in its expressiveness. For example, the concept of a map which has as many mappings as possible cannot be directly expressed, but can only be realized by establishing a map with exactly one mapping, with exactly two mappings, etc., and then seeing what was the maximum number of mappings reached. Another limitation is that maps are described as having some particular pre-specified combination of mappings, rather than a flexible set of mappings. For example, the concept of a map which has any mapping at all cannot be expressed.

Overall, rather than describing maps by such a large number of highly specific hypotheses, it would seem desirable to describe some of their broad characteristics in a more general and flexible manner. This more general description would seem to need a "higher level" approach. Such an approach is developed in the next section, Section 5.2, the use of metalevel justification is investigated.

5.2 Expressing Justification in the Metalevel

In this section, first the metalevel expression of a justification for partial equality is discussed. Following this, procedures are described for such an expression. Examples are then given of analogy problems and their solutions when metalevel justification is used.

5.2.1 Determining Analogical Maps for Justification

5.2.1.1 Adding a Justification for Proving a Hypothesis

A justification is to be proposed for partial equality which is metalevel in nature. Such a justification would be expressed procedurally and added to Theorist's procedures for handling hypotheses. This is the same manner by which consistency justification is expressed.

In Fig. 5.5 are procedures needed for establishing a hypothesis instance, satisfying a consistency justification, and asking for an analogical justification. When an instance of a possible hypothesis is to be proved, Theorist first checks if that instance is already in the current theory (i.e., has already been assumed before). If so, it does not have to be re-established. If not, it is a new hypothesis instance. To assume a new hypothesis instance, Theorist first proves all the subgoals (using the *proveAll* relation). Second, it establishes all the required metalevel justifications. For instances of the partial equality hypothesis, two justifications have been specified: consistency of the theory and an analogical justification. For instances of other hypotheses, the justification consists of the consistency criterion alone.

The analogical justification that has been chosen for implementation is similar to the one that was expressed in the object level in Section 5.1. There are two main differences. The first is that the justification is generalized to allow any number of mappings in a map, rather than a particular specified number. As well, a heuristic for establishing a maximal number of mappings can be expressed. The second difference is that a one-to-one constraint has been on the functors and arguments of the mappings.

The analogical justification that will be shown is a simple one, one which is a basic part of many approaches to analogical inference. As in the object level approach of Section 5.1, the goal is not to suggest new heuristics for establishing a mapping, but rather to express fundamental heuristics in the hypothetical reasoning framework. The analogical justification is described in more detail in Section 5.2.2. Below is Fig. 5.5, which shows the Theorist procedures for establishing a hypothesis.

```
% Reuse a hypothesis instance if it has previously been assumed.
% If it is a new instance of a partial equality hypothesis, establish a
% consistency justification and analogical justification for it. Otherwise,
% justify the new instance by establishing a consistency justification for it.

proveHypothesis(G, A, N, B, H1, H2) :-
        element(N, H1),
        proveAll(B, [G|A], [N|H1], H2).

proveHypothesis(G, A, N, B, H1, H2) :-
        functor(N, Fname, Arity),
        equal(Fname, partial_equality),
        not element(N, H1),
        proveAll(B, [G|A], [N|H1], H2),
        consistency_justific(G, H1),
        analogical_justific(N, A, H1).
```

```
proveHypothesis(G, A, N, B, H1, H2) :-
        not element(N, H1),
        proveAll(B, [G|A], [N|H1], H2),
        consistency_justific(G, H1).

consistency_justific(G, H) :-
        negate(G, NG),
        not proveNH(NG, [], H).

analogical_justific(partial_equality(T,S), A, H) :-
        map(S, T, [], FinalCorr, [], FinalMppg, H),
        writef("%META  JUSTIFICATION is: %p%n", [FinalMppg]).

equal(X, Y) :-
        X==Y.
```

Figure 5.5 Theorist Procedures for Establishing Hypotheses

Note that when the analogical justification is in the metalevel, the partial equality hypothesis is of the following form:

```
hypothesis partial_equality(T, S):
        dom(T, equal_partially(S, T)).
```

The hypothesis has no antecedents, i.e., no object level justification conditions. In particular it does not have the condition that a map(S, T) relation is true, as the partial equality hypothesis in Section 5.1 had.

5.2.1.2 Expression of Mapping Heuristics

Fig 5.6 shows the procedures for a map, mapping, submap, and other aspects of the justification. These procedures are analogous to, respectively, individual map hypotheses, property mapping hypotheses, and the partial equality conditions of property mapping hypotheses. The close similarity in structure and content can be seen the procedures in Fig 5.6 are compared with the object level hypotheses in Section 5.1.2. The main differences between the content, as was mentioned in Section 5.2.1.1, are the ability to express heuristics for a flexible, maximal set of mappings, and for a one-to-one correspondence among mappings.

```
% For analogical justification of a partial equality hypothesis, establish a map
% between source S and target T. Establish a map by establishing as many
% mappings as possible between properties of S and properties of T.

map(S, T, Corr, FinalCorr, Mppg, FinalMppg, H) :-
        proveNH(dom(S, PropS), [], H),
        functor(PropS, FunctorS, __),
        not element(corr_reln(FunctorS, __), Corr),
        proveNH(dom(T, PropT), [], H),
        functor(PropT, FunctorT, __),
        not element(corr_reln(__, FunctorT), Corr),
        mapping(S, T, PropS, PropT, Corr, NewCorr, Mppg, NewMppg, H),
        map(S, T, NewCorr, FinalCorr, NewMppg, FinalMppg, H).
```

```
map(S, T, FinalCorr, FinalCorr, FinalMppg, FinalMppg, H) :-
        not (FinalMppg = []).

% Establish a mapping by establishing there is a correspondence between
% the relation names and the individual arguments of the properties
% Relations have a correspondence if they are equal in name and arity.
% A correspondence (submap) must be established for each pair of
% arguments, or the mapping fails. Successful mappings are recorded.

mapping(S, T, PropS, PropT, Corr, NewCorr, Mppg, NewMppg, H) :-
        functor_corresp(PropS, PropT),
        PropS =.. [FunctorS|IndivlistS],
        PropT =.. [FunctorT|IndivlistT],
        many_submaps(S, T, IndivlistS, IndivlistT, Corr, [], FinalTentC, [], FinalTentM, H),
        append(Corr, [corr_reln(FunctorS, FunctorT)], Intermed),
        append(Intermed, FinalTentC, NewCorr),
        append(Mppg, [mapping(PropS, PropT)], Intermed2),
        append(Intermed2, FinalTentM, NewMppg).

functor_corresp(PropS, PropT) :-
        functor(PropS, FunctorS, ArityS),
        functor(PropT, FunctorT, ArityT),
        equal(FunctorS, FunctorT),
        ArityS =:= ArityT.

equal(X, Y) :-
        X==Y.
```

```
% For a submap to be established between two arguments IndivS
% and IndivT of a property, one of the following cases must
% be true.  1) IndivS = S and Indiv T = T.
%           2) IndivS and IndivT have a previously established correspondence
%           3) IndivS = IndivT.
%           4) IndivS and IndivT successfully have a map established between them

many_submaps(S, T, [], [], Corr, FinalTentC, FinalTentC, FinalTentM, FinalTentM, H).

many_submaps(S, T, [IndivS|OtherIndivS], [IndivT|OtherIndivT], Corr, TentC, FinalTentC,
        TentM, FinalTentM, H) :-
        submap(S, T, IndivS, IndivT, Corr, TentC, NewTentC, TentM, NewTentM, H),
        many_submaps(S, T, OtherIndivS, OtherIndivT, Corr, NewTentC, FinalTentC,
        NewTentM, FinalTentM, H).

submap(S, T, IndivS, IndivT, Corr, TentC, NewTentC, TentM, NewTentM, H) :-
        equal(IndivS, S),
        not equal(IndivT, T),
        fail.

submap(S, T, IndivS, IndivT, Corr, TentC, NewTentC, TentM, NewTentM, H) :-
        equal(IndivT, T),
        not equal(IndivS, S),
        fail.

submap(S, T, IndivS, IndivT, Corr, TentC, NewTentC, TentM, NewTentM, H) :-
        equal(IndivS, S),
        equal(IndivT, T).
```

```
submap(S, T, IndivS, IndivT, Corr, TentC, NewTentC, TentM, NewTentM, H) :-
        element(corr_indiv(IndivS, IndivT), Corr),
        append(TentC, [], NewTentC),
        append(TentM, [], NewTentM).

submap(S, T, IndivS, IndivT, Corr, TentC, NewTentC, TentM, NewTentM, H) :-
        not element(corr_indiv(IndivS, __), Corr),
        not element(corr_indiv(__, IndivT), Corr),
        equal(IndivS, IndivT),
        append(TentC, [corr_indiv(IndivS, IndivT)], NewTentC),
        append(TentM, [], NewTentM).

submap(S, T, IndivS, IndivT, Corr, TentC, NewTentC, TentM, NewTentM, H) :-
        not element(corr_indiv(IndivS, __), Corr),
        not element(corr_indiv(__, IndivT), Corr),
        map(S, T, IndivS, IndivT, [], SubCorr, [], SubMppg, H),
        append(TentC, [corr_indiv(IndivS, IndivT)], Intermed),
        append(Intermed, SubCorr, NewTentC)
        append(TentM, SubMppg, NewTentM).
```

Figure 5.6 Analogical Justification in the Metalevel

For a map relation to be true, the following is involved. First, it is proved that some property is in the source domain. Second, it is checked that the property has not been mapped previously. If it is already part of a mapping, then, in order to preserve the one-to-one constraint, it should not be mapped again. Third, the same is done for some property of the target. Fourth, a test is done to see if a mapping can be established between the source and target properties. Finally, this entire process is repeated until no more mappings can be established for the justification.

For a mapping relation to be true, the following is involved. First, it is seen if there is a correspondence between the two functors of the properties. Second, submaps (partial equalities) are proven for all the arguments (individual constants) of the properties.

The functor correspondence criteria that have been implemented are very simple. The criteria are firstly that the property names are identical, and secondly that the properties are of the same arity. If these two criteria are met, the functors are said to be acceptable for being mapped to each other.

A submap is true under four cases. The first case is that the pair of arguments in question is actually the source and target themselves. If one of the arguments if the source, but the other is not the target, or vice versa, then the submap should fail. The second case is that there has already been a map established for the individuals in question. In this case, the partial equality succeeds without needing to be reproved. The third case is that the two individuals are equal (and have not been previously mapped before). The fourth case is that a new map needs to be established for the individuals (and they have not been previously mapped before). In this third case, the map relation is to be proved recursively (i.e., partial equality is nested).

Two types of knowledge are kept track of throughout the mapping process. The first consists of the established mappings. Mappings are recorded so they can be displayed in Theorist's response. They are displayed as part of the justification, for the benefit of the user. An example of such a display is:

```
Justification is: [mapping(loves(prince_charming, cinderella), loves(romeo, juliet)),
                   mapping(pursues(prince_charming, cinderella), woos(romeo, juliet))].
```

Besides mappings, correspndences between individual constants and functions are also kept track of. For example, the following are the correspondences for the two mappings shown above:

Correspondences are: [corr_reln(loves, loves), corr_indiv(prince_charming, romeo),
corr_indiv(cinderella,juliet), corr_reln(pursues,woos)].

This knowledge is recorded for the purpose of testing one-to-one correspondence. An example of a one-to-one test which uses the shown correspondences is the following. A mapping is attempted between the properties `helps(step-mother, step-sister)` and `helps(nurse, juliet)`. It is seen from the recorded correspondence `corr_indiv(cinderella, juliet)` that juliet has already been mapped. Hence, a correspondence between juliet and step-sister is not allowed; the attempted mapping fails. Note that while there may be some analogies which have many-to-one mappings, or one-to-many mappings, the examples that will be shown do not include these cases. Such cases would add to the complexity of the mapping heuristics.

More details about one-to-one tests and the map, mapping, and submap processes will be provided in the examples in Section 5.2.2.

5.2.2 Examples

Four examples have been implemented to demonstrate the metalevel approach to justifying partial equality. The firs two examples correspond to the simplest and most complex story examples presented in the object level section (Sec. 5.1). The third example is from a domain of physical systems where the target is an atom and the source is a solar system. The fourth example is from a medical domain, where the analogous cases are patients. All the examples have been implemented using Theorist's commands for entering facts and possible hypotheses into a knowledge base, and for posing queries on the knowledge base. The answers shown in the figures for the examples depict what actual output from Theorist would be like in response to the query posed.

5.2.2.1 Story Analogy 5

Story Analogy 5 is shown in Fig. 5.7. It is an example of analogical inference which uses the analogical justification procedures that have been described. This example has the same domain knowledge as Story Analogy 1 in Section 5.1.2.1. (Compare Fig. 5.7 with Fig. 5.1). It differs in its analogical inference knowledge, in particular, in its hypotheses and justifications.

In order to explain the query that the heroine loves the hero in return, a property ascription is hypothesized, which leads to a partial equality being hypothesized. For the partial equality hypothesis instance, Theorist proves all its subgoals (there are none), then performs its consistency justification, then its analogical justification.

The analogical justification requires a map between cstory and rjstory. The first properties which are proved are `loves(hero,heroine)` from the source, and `loves(hero,heroine)` from the target. Both properties are checked against the list of current correspondences to be sure they have not been mapped before. Since the current list of correspondences is empty, there is no problem. It is then seen whether the two properties can form a mapping. The mapping succeeds: the two functors' correspondence is checked; a submap is established for the individuals hero and hero, because they are identical; and a submap is established for heroine and heroine in the same manner.

The possibilities for mapping are not yet exhausted, so more mappings are tried. A mapping between `woos(hero,heroine)` and `die_together(hero,heroine)` is tried next. However, it fails. The next mapping tried is between `woos(hero,heroine)` and `woos(hero,heroine)`. It is established in a similar manner to the first successful mapping. The functors match because they are identical. The submaps between hero and hero, and between heroine and heroine, succeed because they were already established before. Subsequent attempts at other mappings fail (between the loves_in_return and die_together properties, between the meet_at_a_ball and dies_together properties). In total, there are two successful mappings to make up the map.

Notice that the information about the successful mappings is displayed as part of the metalevel justification. This differs from the examples in Section 5.1, which were concerned with object level justification. In those examples, the information about established mappings was part of the theory, rather than part of the metalevel justification. For those cases, the only metalevel justification was consistency.

STORY ANALOGY 5

Knowledge for source cstory (Cinderella Story)

```
fact dom(cstory, loves(hero, heroine)).
fact dom(cstory, woos(hero, heroine)).
fact dom(cstory, loves_in_return(heroine, hero)).
fact dom(cstory, meet_at_a_ball(hero, heroine)).
```

Knowledge for target rjstory (Romeo and Juliet Story)

```
fact dom(rjstory, loves(hero, heroine)).
fact dom(rjstory, die_together(hero, heroine)).
fact dom(rjstory, woos(hero, heroine)).
```

```
QUERY is:  explain dom(rjstory, loves_in_return(heroine, hero)).
```

Knowledge for Analogical Inference

```
hypothesis  property_ascription(T,S):
        dom(T, loves_in_return(At, Bt))
        ←
         dom(S, loves_in_return(As, Bs))
        & dom(T, equal_partially(S, T)).

hypothesis  partial_equality(T,S):
        dom(T, equal_partially(S, T)).
```

```
META JUSTIFICATION is: [mapping(loves(hero, heroine), loves(hero, heroine)),
                        mapping(woos(hero, heroine), woos(hero, heroine))].
META JUSTIFICATION is: [Consistent_Theory].

THEORY is: [property_ascription(rjstory, cstory),
            partial_equality(rjstory, cstory)].

ANSWER is: [dom(rjstory, loves_in_return(heroine, hero))].
```

Figure 5.7 Story Analogy 5

5.2.2.2 Story Analogy 6

Story Analogy 6 is shown in Fig. 5.8. Its domain knowledge is similar to that of Story Analogy 4 in Section 5.1.2.4. (Compare Fig. 5.8 with Fig. 5.4). This example exhibits three levels of partial equalities. The first mapping tried is between `loves(prince_charming,cinderella)` and `loves(romeo,juliet)`. This mapping requires that a submap be established firstly between prince_charming and romeo, and, if this succeeds, secondly between cinderella and juliet. Since these pairs of individuals have not been previously set in correspondence, and since they are not identical, their own properties must in turn be examined and mapped.

For example, a submap between prince_charming and romeo results from their both being brave and charming, and from prince_charming being a prince and romeo being a boy. This last mapping results in turn from a submap between prince and boy which results from their both being males.

Note that the mappings which result from a submap are recorded and added to the mappings already established to justify the current map. For instance, when the mapping between `isa(prince,male)` and `isa(boy,male)` is established, it is added to the current justification for the map between prince and boy. When the map between prince and boy is established the mapping between `isa(prince_charming,prince)` and `isa(romeo,boy)` is established. Then all the subsidiary mappings are added to the justification for the highest level mapping.

A submap between cinderella and juliet is established in a similar manner. The submap consists of one mapping based on their both being lovely.

In total, seven mappings form a map to justify the partial equality between the two stories. This map successfully supports the inference of the relation in the query.

STORY ANALOGY 6

Knowledge for source cstory (Cinderella Story)

```
fact dom(cstory, loves(prince_charming, cinderella)).
fact dom(cstory, woos(prince_charming, cinderella)).
fact dom(cstory, loves_in_return(cinderella, prince_charming)).
fact dom(cstory, meet_at_a_ball(prince_charming, cinderella)).

fact dom(prince_charming, brave(prince_charming)).
fact dom(prince_charming, charming(prince_charming)).
fact dom(prince_charming, isa(prince_charming, prince)).

fact dom(cinderella, lovely(cinderella)).
```

Knowledge for target rjstory (Romeo and Juliet Story)

```
fact dom(rjstory, loves(romeo, juliet)).
fact dom(rjstory, die_together(romeo, juliet)).
fact dom(rjstory, woos(romeo, juliet)).

fact dom(romeo, brave(romeo)).
fact dom(romeo, charming(romeo)).
fact dom(romeo, isa(romeo, boy)).

fact dom(juliet, lovely(juliet)).
```

General Knowledge

```
fact dom(prince, isa(prince, male)).
fact dom(boy, isa(boy, male)).
```

QUERY is: explain dom(rjstory, loves_in_return(juliet, romeo)).

Knowledge for Analogical Inference

```
hypothesis property_ascription(T,S):
    dom(T, loves_in_return(At, Bt))
    ←
     dom(S, loves_in_return(As, Bs))
    & dom(T, equal_partially(S, T)).

hypothesis partial_equality(T,S):
    dom(T, equal_partially(S, T)).
```

```
META JUSTIFICATION is: [mapping(loves(prince_charming, cinderella), loves(romeo, juliet)),
                        mapping(woos(prince_charming, cinderella), woos(romeo, juliet)),
                        mapping(charming(prince_charming), charming(romeo)),
                        mapping(brave(prince_charming), brave(romeo)),
                        mapping(isa(prince_charming, prince), isa(romeo, boy)),
                        mapping(isa(prince_charming, male), isa(romeo, male)),
                        mapping(lovely(cinderella), lovely(juliet))].
META JUSTIFICATION is: [Consistent_Theory].

THEORY is: [property_ascription(rjstory, cstory),
            partial_equality(rjstory, cstory)].

ANSWER is: [dom(rjstory, loves_in_return(juliet, romeo))].
```

Figure 5.8 Story Analogy 6

5.2.2.3 Solar System and Atom Analogy

Now an example is given of analogical inference for a domain different from the story domain. The source domain is knowledge about a solar system, and the target domain is knowledge about an atom (similar to Gentner's solar system example [Gentner 83]). Whereas in the story analogies the individuals of the domains were mainly characters of the stories, in this analogy, the individuals are physical parts, or components, of the two physical systems. The example is based on simple models of the solar system and atom. See Fig. 5.9.

Theorist is asked to explain that the nucleus attracts the electron in the atom. Theorist uses its analogical inference knowledge and procedural justification for analogy to eventually explain the query successfully. The explanation is based on a property ascription from the solar system's property of the sun attracting the planet. Notice that there is no fact directly stating that the sun attracts the planet. However this conclusion is derived from the fact that the sun gravitationally attracts the planet, in combination with the fact from general knowledge which states that gravitationally attracting implies attracting.

The first two successful mappings are between the properties describing the central positions of the sun and nucleus, and the peripheral positions of the planet and electron. These mappings put into correspondence the sun with the nucleus, and the planet with the electron.

One of the mappings tried is between the properties `orbits(planet,sun)` and `spins_about(electron,nucleus)`. Though these two properties have non-identical relation names, they can still be mapped through the use of some of the general knowledge given. In the general knowledge there are two facts which relate the properties of orbiting and spinning about to revolving around. One fact states that if in some domain there is one individual which spins about a second individual, then it is also true that that first individual revolves around the second individual. A similar fact is true for one individual orbiting another. In effect, these pieces of general knowledge relate two different properties to a common, more general property. It is this more general property which can be proved of both the source and target (within the map procedure), and have its functor mapped through an identical match. The resulting mapping is between: `revolves_around(planet,sun)` and `revolves_around(electron,nucleus)`.

While one "level" of implication between properties is shown in this analogy, in other analogies one could have many properties being related. The relationships could form the equivalent of a "hierarchy of relations" [Burstein 86].

Another mapping which is tried is between the properties: `more_massive_than(sun,moon)` and `more_massive_than(nucleus,electron)`. Correspondences are established between more_massive_than and more_massive_than, and between sun and nucleus. They are recorded in a tentative list as they are established. However, when moon and electron are to be mapped, there is a failure since electron has already been mapped, and a one-to-one correspondence must be preserved. Note that at this point in the process, the tentative list of correspondences is discarded; it is not added to the list of correspondences for successful mappings.

The final map established for the justification has four mappings. This justification is shown in Fig. 5.9.

SOLAR SYSTEM AND ATOM ANALOGY

Knowledge for source solsys (solar system)

```
fact dom(solsys, huge(solsys)).
fact dom(solsys, at_center(sun, solsys)).
fact dom(solsys, at_periphery(planet, solsys)).
fact dom(solsys, orbits(planet, sun).
fact dom(solsys, more_massive_than(sun, moon)).
fact dom(solsys, more_massive_than(sun, planet)).
fact dom(solsys, gravitationally_attracts(sun, planet)).
```

Knowledge for target atom (atom)

```
fact dom(atom, tiny(atom)).
fact dom(atom, at_center(nucleus, atom)).
fact dom(atom, at_periphery(electron, atom)).
fact dom(atom, spins_about(electron, nucleus).
fact dom(atom, more_massive_than(nucleus, electron)).
```

General Knowledge

```
fact dom(X, attracts(A,B)) ← dom(X, gravitationally_attracts(A,B)).
fact dom(X, revolves_around(A, B) ← dom(X, spins_about(A, B)).
fact dom(X, revolves_around(A, B) ← dom(X, orbits(A, B)).
```

QUERY is: explain dom(atom, attracts(nucleus, electron)).

Knowledge for Analogical Inference

```
hypothesis property_ascription(T,S):
        dom(T, attracts(At, Bt))
        ←
           dom(S, attracts(As, Bs))
        & dom(T, equal_partially(S, T)).

hypothesis partial_equality(T,S):
        dom(T, equal_partially(S, T)).
```

(continued on next page)

```
META JUSTIFICATION is:
    [mapping(at_center(sun, solsys), at_center(nucleus, atom)),
    mapping(at_periphery(planet, solsys), at_periphery(electron, atom)),
    mapping(revolves_around(planet, sun), revolves_around(electron, nucleus)),
    mapping(more_massive_than(sun, planet), more_massive_than(nucleus, electron))].
META JUSTIFICATION is: [Consistent_Theory].

THEORY is: [partial_equality(atom, solsys),
           property_ascription(atom, solsys)].

ANSWER is: [dom(atom, attracts (nucleus, electron))].
```

Figure 5.9 Solar System and Atom Analogy

5.2.2.4 Medical Analogy

One final example given for analogical inference with metalevel justification involves medical domains. The source is a previous patient that was diagnosed to have a cardiac arrest from certain indications. The target is a current patient. In this example, each domain only has one individual, i.e., its patient. All the medical indications have been represented as properties on these individuals. See Fig. 5.10.

The question is whether the current patient has a heart seizure. Analogical inference is done with respect to a previous case in the same domain (this type of reasoning can be called case-based reasoning).

The property to be ascribed to the current patient is that of a heart seizure. However, this property cannot be proved of the source, and there is no property ascription hypothesis given which ascribes a heart seizure from a heart seizure. However, there is a hypothesis to ascribe a heart seizure from a cardiac arrest. Hence, this hypothesis is used.

In this example, the ascribed property (i.e., heart seizure) is not the same as the property in the source (i.e., cardiac arrest). That is, the ascribed property is not common to the source as well as the target, and the Common Condition, which is part of Greiner's definition of analogical inference [Greiner 85], does not hold. (Note that the Common Condition was not included in the definition of analogical inference in Section 4.3.) It can be seen from this example that ascription is not necessarily restricted to the cases where the Common Condition holds, but can be more general.

The general knowledge in this example is knowledge in the medical area, but is not knowledge specific to either of the cases. The first two facts of the given general knowledge express a relationship between more specific properties and a more common "parent" property. That is, soreness and sharp pain in the chest are types of chest discomfort in general. The last two general knowledge facts express more of a causal relationship between properties. That is, the indications of feeling dizzy or faint are both caused by the common cause of lacking oxygen.

MEDICAL ANALOGY

Knowledge for source prevpatient (previous patient)

```
fact dom(prevpatient, chest_soreness(prevpatient)).
fact dom(prevpatient, feels_dizzy(prevpatient)).
fact dom(prevpatient, sweaty_palms(prevpatient)).
fact dom(prevpatient, arm_feels_heavy(prevpatient)).
fact dom(prevpatient, cardiac_arrest(prevpatient)).
```

Knowledge for target currpatient (current patient)

```
fact dom(currpatient, sweaty_palms(currpatient)).
fact dom(currpatient, sharp_chest_pain(currpatient)).
fact dom(currpatient, feels_faint(currpatient)).
fact dom(currpatient, arthritis(currpatient)).
```

General Knowledge

```
fact dom(X, chest_discomfort(X)) ← dom(X, chest_soreness(X)).
fact dom(X, chest_discomfort(X)) ← dom(X, sharp_chest_pain(X)).
fact dom(X, lacks_oxygen(X)) ← dom(X, feels_dizzy(X)).
fact dom(X, lacks_oxygen(X)) ← dom(X, feels_faint(X)).
```

QUERY is: explain dom(currpatient, heart_seizure()).

Knowledge for Analogical Inference

```
hypothesis property_ascription(T,S):
      dom(T, heart_seizure(T))
      ←
       dom(S, cardiac_arrest(S))
      & dom(T, equal_partially(S,T)).

hypothesis partial_equality(T,S):
      dom(T, equal_partially(S,T)).
```

(continued on next page)

```
META JUSTIFICATION is: [mapping(chest_soreness(prevpatient),
                                chest_soreness(currpatient)),
                        mapping(lacks_oxygen(prevpatient),
                                lacks_oxygen(currpatient)),
                        mapping(sweaty_palms(prevpatient),
                                sweaty_palms(currpatient))].
META JUSTIFICATION is: [Consistent_Theory].

THEORY is: [partial_equality(currpatient, prevpatient),
            property_ascription(currpatient, prevpatient)].

ANSWER is: [dom(currpatient, cardiac_arrest(currpatient))].
```

Figure 5.10 Medical Analogy

5.2.3 Discussion of the Metalevel Approach

From the above examples, it can be seen how a metalevel expression of justification for partial equality can be used for analogical inference in the Theorist framework. In comparison with the object level approach, one advantage of the metalevel approach is efficiency. Since the knowledge base does not contain a large number of specific hypotheses, Theorist is saved the work of searching through them for the particular ones that are relevant to the query.

Another advantage is increased expressiveness. General characteristics of maps, such as a maximal number of mappings and any combination of mappings, can be expressed. Other characteristics of maps, which have not been shown in the examples, some of which are more complex, could also be captured with a procedural approach; for example, such characteristics could include mappings between certain types of properties [Burstein 86, Winston 80], and mappings of higher systematicity [Gentner 83]. Indeed, most of the research on analogical reasoning has taken a procedural approach.

The increased expressiveness directly affects the increased generality of the metalevel approach. The procedures for justifying the partial equality are applicable to explaining many queries. However, the object level hypotheses were so specific as to be applicable only for partial equalities between certain individuals with certain properties.

A disadvantage of the procedural approach concerns the practical issue of useability. Since the heuristics for maps and mappings are expressed as part of the Theorist procedures, a user who queries Theorist cannot see the heuristics being used. As well, in order for changes to be made in the particular style of mapping, changes have to be made to the procedures rather than simply to the knowledge base. These disadvantages are characteristic of any procedural approach as opposed to a declarative approach.

Chapter 6

Conclusions

6.1 Summary

To summarize, this dissertation has demonstrated that analogical inference can be viewed as hypothetical reasoning in the Theorist framework. The basic nature of analogical inference has been investigated. In particular, the aspects of property ascription and partial equality in the inference have been identified and expressed through two key hypotheses. Mapping heuristics have been expressed as justifications for a partial equality hypothesis. Such map-based justification has been implemented in both an object level manner and a metalevel manner. Finally, different examples of analogical inference have been implemented in Theorist and have been run and tested. These examples have been from various problem domains, and have had different mapping heuristics.

6.2 Relation to Other Work

In this section, the relation of the hypothetical reasoning approach to other approaches in analogical reasoning is discussed.

Though Jackson's work [Jackson 86] is also in the Theorist framework, it has had little influence on the approach of this dissertation. A major difference is that the current work has a more general approach. Jackson's specification has hypotheses specific to only one particular type of analogical inference (i.e., abstraction-based and useful-in-solving-a-problem). In contrast, the current specification is intended to express general analogical inference, and hence accommodates different heuristics. Thus, the various heuristics are separated from the core analogical process.

A second difference is that Jackson uses hypotheses to express knowledge specific to a problem domain (i.e. domain abstractions). On the other hand, the current work uses hypotheses which characterize the analogical inference itself more than they characterize the domain. This characterization is done through hypotheses which capture the concept of a property being ascribed based on an analogical relationship.

Another difference between this work and Jackson's work, is that this work has been implemented. That is, it has been demonstrated that the current approach is realizable through implemented examples, whereas Jackson has not implemented his approach, though he gives suggestions for doing so.

In relation to Greiner's work [Greiner 85, Greiner 86], the approach of this dissertation has been influenced by his formalized methodology and logic-based viewpoint. In addition, the characterization of analogical inference as hypothetical inference parallels his characterization of analogical inference as plausible inference. It is an indication of how well suited the Theorist framework is to analogical inference that the standard conditions for hypothetical reasoning are so similar to those Greiner defined for plausible inference (i.e., the conditions that the inference be unknown and consistent). However, the conditions added to define analogical inference in Theorist differ from Greiner's conditions. The former consist of property ascription and partial equality conditions, whereas the latter consists of the common condition. Greiner also suggests abstraction-based and useful heuristics for mapping, whereas the work of this dissertation is not directly concerned with suggesting particular heuristics.

Another difference is that Greiner does not have a fully developed framework for plausible inference such as has been developed for hypothetical reasoning. Beyond giving a definition, Greiner does not give a detailed specification of how the plausible inference process works independent of analogical inference. His concept of plausible inference seems to have been developed as a step towards analogical inference. In contrast, the hypothetical reasoning framework is of central importance to, and has been developed independently of, analogical inference. As a result, the hypothetical reasoning specification has the advantage of encompassing a broader view of analogy.

With respect to Gentner's approach [Gentner 83, Falkenhainer 86], the view of different partial equality relations is similar to her theory of different types of similarity relations. She defines her similarity relations according to different mapping criteria, for example, whether attribute-type or relation-type properties are equal in the target and source domains. The Theorist research also takes the view that different partial equality relations can be defined according to their specific criteria. However, this view does not have a requirement as to what these criteria must necessarily be, but rather incorporates a flexibility for different reasoners to use different criteria.

Of the different mapping heuristics which various researchers have proposed, the ones that have been implemented in examples have been closest to Winston's [Winston 80, Winston 84a, Winston 84b]. They are data-directed (or bottom-up) heuristics, centered around properties of individuals. These heuristics were chosen for implementation because they are relatively simple, and incorporate aspects of mapping basic to many approaches.

6.3 Contributions

At the start of this dissertation, the thesis was put forward that analogical inference can be expressed as hypothetical reasoning. This claim has been supported by the development of appropriate hypotheses for property ascription and partial equality, by the specification of a Theorist definition with suitable conditions, and by the implementation of several examples of analogical inference.

One of the main contributions of the work of this dissertation is that it helps crystallize the plausibility-based nature of analogical inference. This aspect of inference is often implicit in other approaches to analogy. For instance, it may be embodied in the recognition that proposed analogies can be checked for consistency, tested against other knowledge, ranked acccording to various preferences, and debugged for incorrect assumptions. However, the underlying assumptions in the process are rarely made clear or explicit, as is the case in the hypothetical framework.

In the thesis that was put forward, it was proposed that the expression of analogical inference could be done in a clear, general, and implementable manner. Clarity was achieved through the declarative and formal basis of the framework. A logic-based methodology was used to investigate issues such as what types of knowledge are involved (e.g., facts, assumptions, justifications), and what roles are played by them in the inference.

Generality and flexibility were achieved in that the inference mechanism was not built around any particular mapping heuristic. Rather, the specification has the ability to accomodate different heuristics as different justifications. This approach is also intuitive in that different reasoners can use different justifications for making the same assumption, according to individual preference. Indeed, in psychological tests on analogies, there seems to be a wide spectrum in the individual differences between reasoners [Sternberg 77].

The implementability of the Theorist approach has been demonstrated from the working examples. In the case of object level justification, inference was realizable without any modifications to the Theorist inference mechanism. In the case of metalevel justification, it was implemented in a similar manner to consistency justification, as an augmentation to the inference mechanism. In the case of metalevel justification, the inference mechanism was augmented in a manner similar to that for consistency justification.

Besides providing a clear, general, and implementable specification of analogical inference, the hypothetical reasoning approach also contributes a conceptual view of mapping heuristics. Most other work on analogy is procedural in nature and concentrates on various criteria or heuristics for constructing mappings. The hypothetical view of these heuristics is that they are justifications for assuming partial equality relations. The procedures which implement the heuristics are given meaning as metalevel expressions of such justifications. Through this view, the heuristics have a clearly defined role in the overall reasoning process. The hypothetical approach, then, provides a potentially unifying framework for disparate mapping criteria.

In addition to contributing insights to analogical inference, it is hoped that the work described can serve as an additional example of hypothetical reasoning in Theorist. The expression of another type of reasoning in Theorist (besides diagnostic reasoning, default reasoning, etc.) further supports the claim that the framework is a useful general paradigm for many kinds of reasoning. As well, the work of this dissertation specifies one of the most complex justifications for hypotheses to date. Besides being rather complex, the map-based justification, when expressed in the metalevel, is an example of a local metalevel justification. Few local justifications (rather than global justifications such as consistency) have been previously specified in the metalevel.

6.4 Future Work

There are several areas for future work that are possible. One area (mentioned in Sec. 6.2), is specifying other justifications besides the object-oriented, data-directed heuristics that are demonstrated in the examples shown previously. Other possible heuristics include those which use abstractions [Greiner 86], causal structures [Burstein 86], or other constructs in a goal-directed approach. One issue that would need to be decided for such heuristics is whether to express the goal-directing structures as object level or metalevel relations. If, for instance, the structures are perceived to represent knowledge about the conceptual domain (e.g., laws of physics and medical principles [Greiner 85]), then they might be best represented as object level relations. However, since they are likely higher order relations, they might for practicality and efficiency be represented as metalevel relations.

The use of relevancy criteria is another variation of heuristic which could be investigated. Such criteria emphasize that the analogical map should somehow be relevant to the goal. One such heuristic advocates mapping between causal relations, as opposed to other types of relations, since causation is a form of relevancy [Winston 80]. Since this heuristic describes a category of relations, it might be best represented by using metalevel relations. Hence the justification would be a metalevel one.

Another area for future work is in theory preference. Whereas the criteria expressed in a justification determine valid or acceptable theories, ranking, or preferring, criteria determine preferred theories. Preference criteria would compare valid theories after they are established. There may be overlap between criteria used to justify a theory and criteria used to prefer between theories. This overlap parallels the one between the elaboration and verification of an analogy [Hall 87].

In the example called Story Analogy 1, in Fig. 5.1 (Sec. 5.1.2.1), the use of different possible hypotheses for mappings resulted in more than one theory in the answer. This is the type of situation where preference criteria would be used. For example, the third theory displayed in Fig. 5.1 has two instances of property mapping hypotheses included in it, whereas the other two theories only have one. The third theory would be preferred if there was a preference criteria for theories with more instances of mapping hypotheses. This type of preference criteria could reflect the heuristic that a analogy with more mappings is a better supported analogy.

Examples of theory preference criteria include systematicity (prefer maps with higher systematicity, i.e., containing higher degree relations) [Gentner 83], and minimality (prefer the minimum number of mappings that are able to lead to the solution of a problem) [Greiner 85]. The expression of a criteria such as minimality could be done in the metalevel, using metalevel relations which have the theories themselves as arguments. For example:

```
minimal(Theory1, Theory2) <-- minimality_condition1(_, _)
                              & minimality_condition2(_, _), etc.
```

While it is possible to express arbitrary theory preference criteria as procedures in Theorist, it would be important to develop them in a principled manner, with desirable semantic properties [Goebel 88b].

A third area for future work is the investigation of analogical inference of a predictive kind rather than an explanatory kind. As was discussed in Sec. 1.2, in the explanatory case of analogical inference a goal relation is given, and then an analogy is hypothesized in order to explain it. A different problem occurs when there is no goal to be explained, but an analogy is hypothesized and all its logical consequences, or predictions, are examined. This type of analogical inference is not "query directed" but is predictive in nature. One example of this type of reasoning occurs when a reasoner is presented with a metaphor and must draw inferences from it in order to "comprehend" or analyse the metaphor. Prediction in general, as opposed to explanation in general, has been investigated in the Theorist framework. This other direction of reasoning in the hypothetical framework could potentially capture another direction in analogical inference.

A fourth direction for future work is that of expressing other stages of analogical reasoning through other stages of Theorist's scientific reasoning cycle. For example, the dynamic creation of possible hypotheses (rather than the use of static ones that have been given) could reflect the dynamic creation of mapping strategies. As well, the stages of testing and revision of theories could correspond to the verification and debugging of analogies. However, at this point in time in Theorist, the investigation of these stages in general is not very advanced.

A fifth possibility for investigation is that of integrating analogical reasoning with other types of reasoning. The Theorist framework is general enough to be well suited to this task. For example, hypotheses for medical diagnosis could be used together with hypotheses for analogical inference, such that the two types of reasoning interact with one another.

As has been indicated, there are many possible directions of future work which can follow from the view of analogical reasoning as hypothetical reasoning.

6.5 Conclusion

In conclusion, it is hoped that the work that has been done helps shed insight into the difficult problem of analogical reasoning. There are so many definitions of analogical inference that at times it may seem to be amorphous in nature. However, an attempt has been made to provide a clearer baseline for analogy by delineating its assumption-based nature. This delineation has been done by explicating two essential assumptions which underly analogical inference, assumptions of property ascription and partial equality. Variations in inference can then be phrased as differences in the justification of these assumptions. It has been seen how some examples of this can be done in Theorist. Through such examples and their underlying framework, the way has been paved for future advances in viewing analogical inference as hypothetical reasoning.

Bibliography

[Bowen 85]
K. A. Bowen, Meta-level programming and knowledge representation, *New Generation Computing*, 3 (1985), OHMSHA, LTD. and Springer-Verlag, pp.359-383.

[Burstein 86]
M. H. Burstein (1986), Concept formation by incremental analogical reasoning and debugging, *Machine Learning: An Artificial Intelligence Approach*, R. S. Michalski, J. G. Carbonell and T. M. Mitchell (eds.), Morgan-Kaufmann, Los Altos, California, pp. 351-369.

[Carbonell 83]
J. G. Carbonell (1983), Learning by analogy: Formulating and generalizing plans from past experience, *Machine Learning - An Artificial Intelligence Approach*, R. S. Michalski, J. G. Carbonell and T. M. Mitchell (eds.), pp. 137-161, Tioga, California.

[Carbonell 85]
J. G. Carbonell and S. Minton, Metaphor and Commonsense Reasoning (1985), *Formal Theories of the Commonsense World*, J. R. Hobbs and R. C. Moore (eds.), Ablex Publishing, New Jersey, pp.405-426.

[Falkenhainer 86]
B. Falkenhainer, K. D. Forbus, and D. Gentner (1986), The Structure-Mapping Engine, *Proceedings of the National Conference on Artificial Intelligence (AAAI-86)*, Philadelphia, Pennsylvania.

[Genesereth 87]
M. R. Genesereth and N. J. Nilsson (1987), *Logical Foundations of Artificial Intelligence*, Morgan Kaufman, Los Altos, California.

[Gentner 83]
D. Gentner (1983), Structure-mapping: a theoretical framework for analogy, *Cognitive Science* 7 (2), pp.155-170.

[Gick 83]
M. L. Gick and K. J. Holyoak (1983), Schema induction and analogical transfer, *Cognitive Psychology* 15, pp.1-38.

[Goebel 87]
R. F. Goebel and S. D. Goodwin (1987), Applying theory formation to the planning problem, *The frame problem in artificial intelligence : proceedings of the 1987 workshop*, April 13-15, University of Kansas, Lawrence, Kansas, F. M. Brown (ed.), pp.207-232.

[Goebel 88a]
R. Goebel (1988), Knowledge base prototyping using Prolog tools, *Simposium International de Intelligencia Artificial*, August, Monterrey, Mexico.

[Goebel 88b]
R. Goebel (1988), A Hypothetical Reasoning-Based Specification of Analogical Reasoning, forthcoming.

[Goodwin 87]
S. D. Goodwin (1987), *Representing frame axioms as defaults*, Research report CS-87-48, July 1987, Department of Computer Science, University of Waterloo, Waterloo.

[Greiner 85]
R. Greiner (1985), *Learning by understanding analogies*, PhD. thesis, Stanford University, Stanford.

[Greiner 86]
R. Greiner (1986), *Learning by understanding analogies*, Technical report CSRI-188, University of Toronto, Toronto.

[Hall 87]
R. P. Hall (1987), *Computational approaches to analogical reasoning: a comparative analysis*, Department of Information and Computer Science, Unversity of California, Irvine.

[Haraguchi 86]
M. Haraguchi and S. Arikawa, Reasoning by analogy as a partial identity between models, *Analogical and Inductive Inference*, International Workshop All '86 Proceedings, Wendisch-Reitz, GDR, October 6-10 1986, K.P. Jantke (ed.), Springer-Verlag Lecture Notes in Computer Science, G. Goos, J. Hartmanis (eds.), Springer-Verlag, pp.61-87.

[Holyoak 84a]
K. J. Holyoak (1984), Analogical thinking and human intelligence, *Advances in The Psychology of Human Intelligence*, vol. 2, R. J. Sternberg (ed.), Lawrence Erlbaum Associates, New Jersey.

[Holyoak 84b]
K. J. Holyoak (1984), Mental models in problem solving, *Tutorials in Learning and Memory*, J. R. Anderson, S. M. Kosslyn (eds.), W.H. Freeman, San Francisco, pp.193-218.

[Jackson 86]
W. K. Jackson (1986), *A theory formation framework for learning by analogy*. M.Sc. thesis, Computer Science, University of Waterloo, Waterloo, Ontario.

[Kedar-Cabelli 85a]
S. T. Kedar-Cabelli (1985), *Analogy - from a unified perspective*, Technical report ML-TR-3, Laboratory for Computer Science Research, Rutgers University, New Jersey.

[Kedar-Cabelli 85b]
S. Kedar-Cabelli (1985), *Purpose-directed analogy*, Technical report ML-TR-1, Laboratory for Computer Science Research, Rutgers University, New Jersey.

[Kolodner 85]
J. L. Kolodner (1985), A process model of cased-based reasoning in problem solving, *Proceedings of the Ninth International Joint Conference on Artificial Intelligence (IJCAI-85)*, August 16-18, UCLA, Los Angeles, California, pp.284-290.

[Moore 74]
J. Moore and A. Newell (1974), How can Merlin understand? *Knowledge and Cognition*, L. W. Gregg (ed.), Lawrence Erlbaum Associates, Md., pp. 201-252.

[Poole 86]
D. Poole, R. Goebel, and R. Aleliunas (1986), *Theorist: a logical reasoning system for defaults and diagnosis*, Research report CS-86-06, Logic Programming and Artificial Intelligence Group, Department of Computer Science, University of Waterloo, Waterloo.

[Poole 87]
D. L. Poole (1987), *Defaults and conjectures: hypothetical reasoning for explanation and prediction*, Research report CS-87-54, Logic Programming and Artificial Intelligence Group, Department of Computer Science, University of Waterloo, Waterloo.

[Rumelhart 81]
D. E. Rumelhart and D. A. Norman (1981), Analogical processes in learning, *Cognitive Skills and Their Acquisition*, J.R. Anderson (ed.), Lawrence Erlbaum Associates, New Jersey, pp. 335-359.

[Sternberg 77]
R. J. Sternberg (1977), Theories of Analogical Reasoning: A Literature Review, Chapter 5 of *Intelligence, Information Processing, and Analogical Reasoning: The Componential Analysis of Human Abilities*, Lawrence Erlbaum Associates, New Jersey, pp.99-133.

[Tversky 77]
A. Tversky, Features of Similarity (1977), *Psychological Review*, 84(4), pp. 327-352.

[Watanabe 87]
L. M. Watanabe, *Guiding constructive induction for incremental concept learning from examples*, M.Sc. thesis, Deparment of Computing Science, University of Alberta, Edmonton, Alberta.

[Wellsch 85]
K. C. Wellsch (1985), *A computational model for reasoning by analogy*, M.Sc. thesis, Department of Computer Science, University of Waterloo, Waterloo, Ontario.

[Winston 78]
P. H. Winston (1978), Learning by creating and justifying transfer frames, *Artificial Intelligence*, vol. 10, North-Holland, pp.147-172.

[Winston 80]
P. H. Winston (1980), Learning and reasoning by analogy, *Communications of the ACM*, 23(12), pp.689-703.

[Winston 82]
P. H. Winston (1982), Learning new principles from precedents and exercises, *Artificial Intelligence*, vol. 19, North-Holland, pp.321-350.

[Winston 84a]
P. H. Winston (1984), Learning rules from experience, *Artificial Intelligence (2nd Ed.)*, Addison-Wesley, pp.415-438.

[Winston 84b]
P. H. Winston (1984), Description matching and goal reduction, *Artificial Intelligence (2nd Ed.)*, Addison-Wesley, pp.21-43.

Appendix

The Theorist Interpreter

```
%**********************************************************************
%
%           Module:     Module named explain.p which is the Theorist interpreter.
%
%           Version :   Version shown includes revisions as of Oct. 4, 1988
%                       by George Ferguson. (Note that the version shown
%                       includes the facility to handle a class of knowledge called
%                       askables, but askables are not used by analogical inference.)
%
%**********************************************************************


%
% explain(G) is always true and interfaces with explain/2
%

explain(G) :-
        explain(G,H),
        nl,
        writef("Theory is: %p%n%n",[H]),
        writef("Answer is:%p%n",[G]),
        nl,
        fail.

explain(_) :-
        write('No (more) answers'),
        nl,
        !,


%
% explain(G,H) :        True when G is a list of goals which can be explained
%                       using the facts and the consistent set of hypotheses H.
%

explain(G,H) : -
        proveAll(G,[],[],H).
```

```
%
% prove(G,A,H1,H2) :        True when the atomic goal G is proved with
%                           ancestors A and hypotheses H1+H2.
%

prove(G,A,H,H) :-                          % proof by contradiction
        negate(G,NG),
        member(NG,A).

prove(G,A,H1,H2) :-                              % fact expands proof tree
        fact(G,B),
        proveAll(B,[G|A},H1,H2).

prove(G,_,H,H) :-                                % askable was answered true
        askable(G),
        wasAsked(G),
        answerWasTrue(G).

prove(G,A,H,H) :-                          % askable can't be deduced and is
        askable(G),                        % asked
        negate(G,NG),
        not wasAsked(G),
        not wasAsked(NG),
        not proveNA(G,A,H),
        not proveNA(NG,A,H),
        askableIsTrue(G).

prove(G,_,H,H) :-                                % meta evaluates to true
        meta(G),
        call(G).

prove(G,A,H1,H2) :-                              % else need hypothesis
        hypothesis(N,G,B),
        proveHypothesis(G,A,N,B,H1,H2).


%
% proveAll(L,A,H1,H2) :     True when all elements in the list of goals L
%                                  have been proved with ancestors A and
%                                  hypothesis H1+H2.
%

proveAll([],_,H,H).

proveAll([G|L],A,H1,H3) :-
        prove(G,A,H1,H2),
        proveAll(L,A,H2,H3).
```

```
%
% proveHypothesis(G,A,N,B,H1,H2) :      True when goal G is proved with
%                                               ancestors A using hypothesis named
%                                               N which has body B, together with
%                                               previous hypothesis H1+H2.
%

proveHypothesis(G,A,N,B,H1,H2) :-                    % reuse previous hypothesis
        element(N,H1),
        proveAll(B,[G|A],H1,H2).

proveHypothesis(G,A,N,B,H1,H2) :-                    % use new hypothesis
        not element(N,H1),
        proveAll(B,[G|A],[N|H1],H2),
        consistency_justific(G,H1).


%
% consistency_justific(G,H) :       True if fail to show the negation of G is true
%                                   using hypothesis H.
%

consistency_justific(G,H) :-
        negate(G,NG),
        not proveNH(NG,[],H).


%
%
% proveNH(G,A,H) :          True if G is proven with ancestor A and hypothesis H
%                           (as prove/4 above but no new hypothesis can be
%                           added). Used to establish consistency.
%

proveNH(G,A,_) :-                        % proof by contradiction
        negate(G,NG),
        member(NG,A).

proveNH(G,A,H) :-                        % fact expands proof tree
        fact(G,B),
        proveAllNH(B,[G|A],H),

proveNH(G,A,H) :-                        % use hypothesis of current theory
        hypothesis(N,G,_),
        element(N,H).

proveNH(G,_,_) :-                        % meta evaluates to true
        meta(G),
        call(G).

proveNH(G,_,H) :-                        % askable was answered true
        askable(G),
        wasAsked(G),
        answerWasTrue(G).
```

```
proveNH(G,A,H) :-                         % askable can't be deduced and is
        askable(G),                       % asked
        negate(G,NG),
        not wasAsked(G),
        not wasAsked(NG),
        not proveNA(G,A,H),
        not proveNA(NG,A,H),
        askableIsTrue(G).


%
% proveAllNH(L,A,H) :       True if all the goals in L can be proved without
%                           introducing new hypotheses as in proveNH/2.
%

proveAllNH([],_,_).

proveAllNH([G|L,A,H) :-
        proveNH(G,A,H),
        proveAllNH(L,A,H).


%
% proveNA(G,A,H) :          True if G is proven with ancestors A and hypotheses
%                           H (as prove/4 above but no new hypotheses or
%                           askables can be used). Used to try and deduce the
%                           value of an askable without asking it.
%

proveNA(G,A,_) :-                         % proof by contradiction
        negate(G,NG),
        member(NG,A).

proveNA(G,A,H) :-                         % use hypothesis of current theory
        hypothesis(N,G,_),
        element(N,H).

proveNA(G,A,H) :-                         % fact expands proof tree
        fact(G,B),
        proveAllNA(B,[G|A],H).

proveNA(G,_,_) :-                         % meta evaluates to true
        meta(G),
        call(G).

proveNA(G,A,H) :-                         % user answered true already
        askable(G),
        wasAsked(G),
        answerWasTrue(G).
```

```
%
% proveAllNA(L,A,H) :        True if all the goals in L can be proved without
%                            introducing new hypotheses or asking new
%                            questions as in proveNA/3.
%

proveAllNA([],_,_).

proveAllNA([G|L],A,H) :-
        proveNA(G,A,H),
        proveAllNA(L,A,H).


%
% negate(G,NG) :     True when NG is the Theorist-negation of G, n(G).
%

negate($NOT(G),G) :-
        not G = $NOT(_).

negate(G,$NOT(G)) :-
        not G = $NOT(_).
```