Calvin: A System for Automating Cosmogenic Isotope Dating

Laura Rassbach

University of Colorado at Boulder

Technical Report CU-CS 1040-08

March 2008

1. Introduction

Scientific reasoning is a complex process, alternately requiring flashes of insight and tedious analysis. This dichotomy is evident in constructing a geologic timeline for a landform using cosmogenic isotope dating. Experts in this field frequently spend months on repetitive mathematical tasks, until they have gathered enough information to suddenly understand the data. The Calvin project[1] is aimed at understanding and automating this process, particularly the semi-mechanical analysis tasks that must be performed on every new problem instance. Having this analysis done quickly and automatically will free experts for the more difficult and creative tasks in the field. Furthermore, it is my hope that Calvin's architecture can eventually be applied to other fields with similar challenges.

The aims of the project are challenging because the science of cosmogenic isotope dating is quite new. There is only a relatively small number of completed, detailed analyses from which one can draw knowledge. Therefore, it is necessary to build a knowledge base through interaction with experts. Unfortunately, as is often the case, it is difficult for these experts to clearly articulate how and why they come to specific conclusions. I have been working with experts in this field for more than two years, but the knowledge base I have so far developed is still very incomplete. Complicating the difficulty of acquiring the knowledge used by experts, few theories in the area are completely formed or fully understood. As a result, most expert analysis relies on vague heuristics that are frequently contradictory. Finally, as is common in most scientific fields, observed data is typically noisy, often sparse, and not completely trustworthy. Any system that automates the timeline construction process, then, must be able to handle contradiction, uncertain heuristics, and untrustworthy inputs gracefully.

Argumentation is an elegant and intuitive framework for automating this quixotic type of reasoning. It is a symbolic logic, which has a number of advantages. In particular, rules acquired from experts can be input directly into the system in a format that closely mirrors their original form, and answers generated by the system can be justified using human-readable explanations. Presenting the system's reasoning to the user in a legible format facilitates engagement and helps to speed further knowledge engineering. In the argumentation architecture used by Calvin, conclusions can receive partial support (modeling uncertain

1.Named for the character in the Calvin & Hobbes comics, who often starts arguments just for the sake of having an argument.

heuristics), and support for a conclusion can be defeated by contrasting evidence or rules (handling contradiction gracefully).

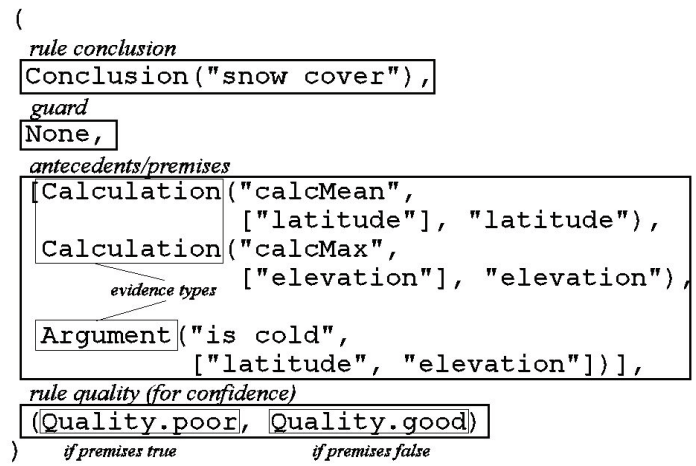

Figure 1: Anatomy of a Calvin rule. This rule expresses the idea that when an area is not cold, snow cover is a much less likely explanation (when the area is cold, snow cover is a slightly more likely explanation for a sample set).

Figure 1 shows an example rule from Calvin. The architecture is discussed in more detail in Section 7; this short introduction is intended to provide a basis for discussion. As with most symbolic AI systems, Calvin's knowledge base is encapsulated in rules. These rules are implicitly Horn clauses, where the conclusion is implied by the conjunction of the premises. In addition to these usual features, a Calvin rule may have a guard that prevents the rule from being used in inappropriate circumstances. The rule in Figure 1, for instance, can only be used when the landform under consideration is a moraine. Finally, because Calvin is an argumentation system, its rules are not absolute. All evidence (including arguments) has a confidence value so that the relative strengths of arguments can be compared. Every rule contains instructions for how to combine the confidence in its premises to obtain a confidence for its conclusion.

This proposal presents the prototype version of the Calvin system along with some preliminary results, and describes how I will complete this thesis. My current results, though sparse, show significant initial success in accurately modeling the reasoning process of isotope dating experts. Section 2 details the process of constructing a timeline for a landform using cosmogenic isotope dating. Section 3 discusses the particular challenges that arise in attempting to automate the type of quixotic reasoning used by experts in this field. Section 4 presents specific architectures that I have rejected as unsuitable for the problem. Section 5 advances argumentation as an elegant and intuitive solution to these challenges. Section 6 covers related work not addressed in Section 4. Section 7 elaborates on the Calvin architecture. Section 8 walks through concrete examples of the working prototype. Section 9 summarizes and presents my plan of work.

2. Cosmogenic Isotope Dating

2.1. What is isotope dating?

Cosmogenic isotope dating is a method for computing the age of a landform using radioactive isotope measurements of samples taken from that landform. This dating procedure is based on the knowledge that cosmic rays hit the earth at a fairly constant rate. When these rays come into contact with certain naturally occurring isotopes, they can cause a nuclear reaction that forms specific isotopes that must be formed by nuclear reactions (e.g. Chlorine-36 or Aluminum-26). Like with the process of radioactive decay, these isotopes are created at a calculable rate. Most types of cosmic rays penetrate only a few inches, so these cosmogenic isotopes are generated almost exclusively at the surface. This knowledge enables a geologist to determine how long a particular sample has been at the surface based on the number of cosmogenic isotopes present and the sample's chemistry.

Unlike the process of radioactive decay, however, the production rate of new isotopes varies over time, significantly, due to a number of factors. These include changes in the level of background cosmic radiation, changes in the location of the earth's magnetic pole, and variations in sea level. As a result, the mathematics involved are quite complicated. Currently these complex calculations are performed in Microsoft Excel; the ACE project (Anderson et al. 2007) is aimed at developing a dynamic application to perform these calculations in a more standardized and structured way. Calvin will eventually be integrated with ACE, since it is intended to reason about samples whose ages have already been calculated by ACE.

For many landforms, the length of time that a sample has been at the surface is actually a measure of the age of the landform. Moraines are an excellent example of this. When a glacier forms a glacial valley, it carves rock from deep underneath the original surface. This rock then becomes part of the glacier, along with soil and other debris. When the glacier begins to melt and retreat back up the valley, it deposits this soil and rock in one or more moraines. Because the boulders were shielded from cosmic rays by the rock above them, and later by the ice of the glacier itself, they are first exposed when the moraine is formed. Thus the surface exposure times of the boulders in a moraine are good measures of when the moraine was actually formed. One common and important application of isotope dating is to construct a complete timeline for a glacier using the ages of the moraines it deposited and samples from the walls of the glacial valley.

Although cosmogenic isotope dating is used primarily to estimate the ages of suddenly-created landforms, it can also yield information about landforms formed over a longer period of time, or from rock that was at the surface prior to the landform's formation. One example of such a landform is an alluvial fan, which is formed over many years by soil deposits from a river, typically at the point where it flows into the sea. Information about the range of sample ages in such a fan can help a scientist understand both when it began to form and how long the formation took.

2.2. The Process of Isotope Dating

An expert who wants to determine the age of a landform using cosmogenic isotope dating will begin by taking samples of surface rock (generally thin chips from several boulders) from the landform. How many samples he/she takes is often completely determined by the number of surfaces that are adequate to sample. If there are many surfaces suitable for sampling, he/she will generally take as many samples as possible, although the number of samples actually analyzed may be limited by financial constraints. Significant expertise is needed to choose good samples—those whose apparent ages are most likely to be the age of the landform: sample boulders should not be excessively weathered, should show no signs of having been rolled or turned, should usually be of similar composition to the surrounding surface, etc. In many cases it will only be possible to take a small number of samples that meet these requirements. I have seen a number of example studies performed with only three or four samples, simply because no more acceptable samples were available. When taking samples, experts record as much data as possible about the location and status of the samples. This includes the sizes of the boulders they are collected from, the amount of visible sky, and the exact location where the sample was taken. Anything unusual about the sample site (such as a weathered boulder, or distance from/proximity to other samples) is especially important to record. Knowing that a sampled boulder had significant signs of weathering, for example, allows experts to correct for erosion in their calculations from the outset.

After collection, the expert sends the samples to an accelerator mass spectrometry (AMS) lab that measures the chemical properties of the samples, specifically the chemical composition and the percentage of the isotope of interest for dating (e.g. Chlorine-36 compared to overall Chlorine). The lab's report includes these numbers with a margin of error for each item. These services are extremely costly, further limiting the number of samples for which data are available.

Using the AMS measurements, information about where the sample was taken, and a large (5-10 sets of data of 300-400 datapoints each) amount of background knowledge, the expert calculates preliminary (or "apparent") ages for the samples. There are three major classes of background knowledge: "worldwide" data, localized information, and chemical information. "Worldwide" data is general information about changes that affect the number of cosmic rays hitting the earth during any given period, such as background radiation information and sea level. Localized information includes the amount of visible sky and local snowfall, which affect the exact number of cosmic rays hitting any particular sample site. Finally, knowledge about chemistry is required to understand the odds of the isotope in question being formed when a cosmic ray hits the sample (including secondary production from chain reactions) (Lal 1958), (Gosse and Phillips 2001). For some of the "worldwide" data, multiple measurements are available and will yield slightly different results. These datasets are being collected and collated as part of the CRONUS Earth project (CRONUS 2007). Handling these calculations and organizing background data is the task of the ACE project (Anderson et al. 2007). Although Calvin is involved only after these initial ages are calculated, it is critical that the system have knowledge about the subtleties that can affect initial ages in complex ways, as I will show in the next section.

2.3. Deriving a Landform Age

After obtaining an exposure age for each sample taken from a landform, the expert needs a single age (or age range) for the sampled landform, with a small enough margin of error that the age is trustworthy. After running the ACE system on the chemical data from the AMS lab, all he/she knows is an age for each individual sample, with individual margins of error calculated based on the reported margin of error from the AMS process. The expert's next step is to calculate the average age and the average margin of error *over the entire sample set*. Using the age range specified by these two averages (average age +/- average error), he/she now checks that all of the calculated sample ages for this landform fall inside this range. If so, then the age of the landform is the average sample age, +/- the average error.

Unfortunately, this happy situation rarely occurs. Instead, it is much more common that the initial sample ages will be spread over a much wider range than that defined by the two sample averages, often as much as 10,000 years (Shanahan and Zreda 2000). In this case, it is necessary to construct an explanation for the spread in apparent ages, usually involving a geologic process acting on the sample set. Once a process that explains the data has been

found, experts can use further calculations to remove its effects from the sample set and (hopefully) arrive at a single age for the landform.

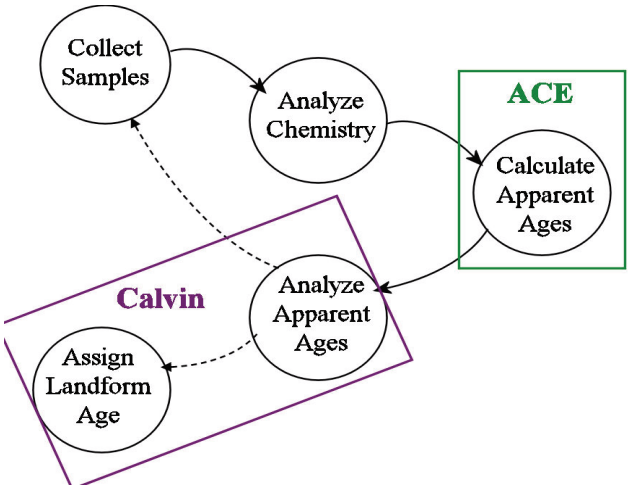

Figure 2: The cyclic process of cosmogenic isotope dating. Portions of this process automated by ACE and Calvin are outlined.

Even this slightly less happy situation—ending with a single explanation after only one round of sampling—is relatively unusual. Often there is no good explanation for the data, or there are several explanations that explain the data equally well. Occasionally the final explanation will involve a significant leap of insight resulting in a completely novel explanation. Generally, the result of the first round of analysis is the conclusion that more samples are needed. This leads to a trip back to the original site to collect more samples with the specific questions left by the first analysis in mind. Guided by the first round of analysis, experts can focus on samples more likely to determine which candidate process is responsible for the skew in the data. For example, if the landform is a moraine and both inheritance (previous exposure) and erosion of the matrix (gradual exposure of previously buried boulders) are suspected, a soil sample can generally distinguish between these two processes. If the experts suspect that they have inadvertently sampled two landforms, they might take several more samples physically close to each of their initial samples, both to clearly distinguish the boundaries between the two landforms and to confirm the initial results (by ruling out lab error and mistaken sample identification). In those cases where experts suspect an unusual explanation will best fit their data, they must completely rule out other, better-understood processes before a novel explanation can be accepted by their research community. This usually requires a larger number of samples and a statistical demonstration that the more common processes are extremely unlikely to have produced the calculated set of ages. Figure 2 illustrates this cyclic process.

The purpose of the Calvin system is to automate the process described in the previous two paragraphs: finding an appropriate explanation for the age spread in a given set of samples. In order to do this I have focussed primarily on reproducing the heuristic reasoning used by experts to determine what processes are most likely, given an initial set of sample ages and other knowledge about the landform in question. The general knowledge and observations used in this process are heterogeneous, from detailed numeric information about a sample's

chemistry to a general observation that a boulder showed signs of weathering. In addition, as discussed in Section 3, the heuristics used by experts are traditionally difficult issues for AI systems.

3. Quixotic Reasoning

Most explanations for spread in apparent ages used by experts come from a list of about fifteen geologic processes that affect the actual exposure times of samples from a single landform. The two most common of these are erosion and inheritance[2]. Erosion is a process that gradually exposes new surfaces, so that the apparent age of a sample is younger than the whole landform (i.e. younger than the length of time that the landform's surface has been exposed). Inheritance is a term for situations where landforms are formed from rock that had been previously exposed at the surface. For example, a moraine might contain boulders that were not carved from far enough below the surface to be wholly unexposed to cosmic rays before the formation of the moraine. Inheritance causes the apparent ages of samples to be older than the actual age of the landform. Other processes that may explain apparent age spread include cover from snow or vegetation, gradual formation such as from soil deposits, or earthquakes, which may suddenly expose large amounts of rock at the surface. Statistical "processes" may also explain the data from a particular landform: perhaps the age spread is a result of lab error or some form of mis-sampling.

Despite the relatively small number of candidate processes, selecting an explanation for the apparent age spread of a particular landform is not a simple task. Available data are noisy and may not be trustworthy. Experts have been known to mis-identify the kind of landform under consideration, or to sample two landforms believing they were a single landform. Multiple processes may have acted on the samples. The manifestation of one process may be quite similar to the manifestations of other processes. For instance, gradual formation and erosion result in data that appear very similar. Finally, it is important not to forget that some explanations do not come from this set list, and instead require a leap of creative insight.

2.These two processes are so common that even when all samples from a landform have the same apparent age they are often considered as a matter of general procedure.

3.1. Mathematical Models

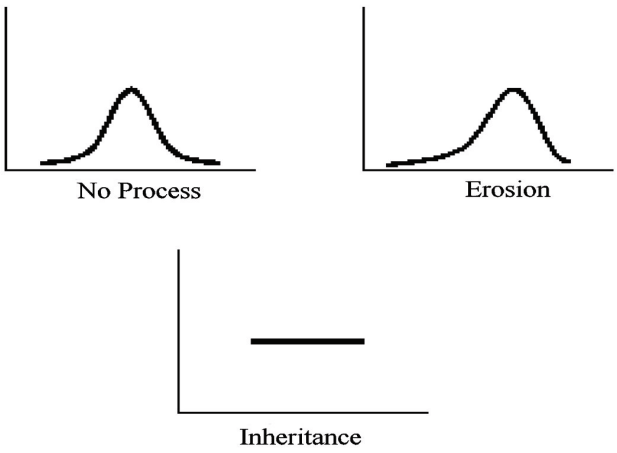

Figure 3: Distribution of sample age (x) vs. number of samples (y) expected for various geologic processes.

Most of the processes that affect apparent ages of samples are expected to result in a characteristic distribution of apparent ages vs. number of samples. For example, erosion looks something like a skewed bell curve with the peak towards the older end of the scale. Inheritance, on the other hand, usually involves a simple uniform distribution over an age range[3]. Figure 3 shows some examples of these distributions. Unfortunately, it is difficult to diagnose the process that affected a particular landform from the distribution of apparent ages because one rarely has enough samples to see the distribution shape. Furthermore, these expected distribution shapes are less useful than one might expect because the mathematical models that produce them are generally inaccurate or incomplete.

The model for inheritance presupposes that every sample was exposed to some cosmic rays before coming to rest in the landform of interest. This exposure is expected to be distributed evenly over a closed range of sample ages. When this is true, a set of samples with a single age (or Gaussian age distribution, after error) will be spread out to an approximately linear distribution. However, it is certainly possible that all samples in a landform received about the same amount of previous exposure, or that some samples have significant inheritance and others none. If the boulders sampled in a moraine originally came from a fault scarp, for example, then the amount of inheritance would be in several distinct clusters, rather than a uniform distribution. The simple expectation of a uniform or approximately uniform distribution fails to capture these complexities.

From discussions with experts, it appears that there are indeed multiple types of inheritance, but this distinction has been extremely difficult to isolate. When discussing inheritance in abstract, experts will generally predict a simple linear distribution of ages among a set of samples subject to inheritance. (In fact, the experts that I am consulting with are currently working on a *combined* model of inheritance and erosion that uses this assumption.) However, when discussing particular landforms, “inheritance” is considered a good explanation when

3.The distribution is actually a Gaussian that is so spread out it appears uniform

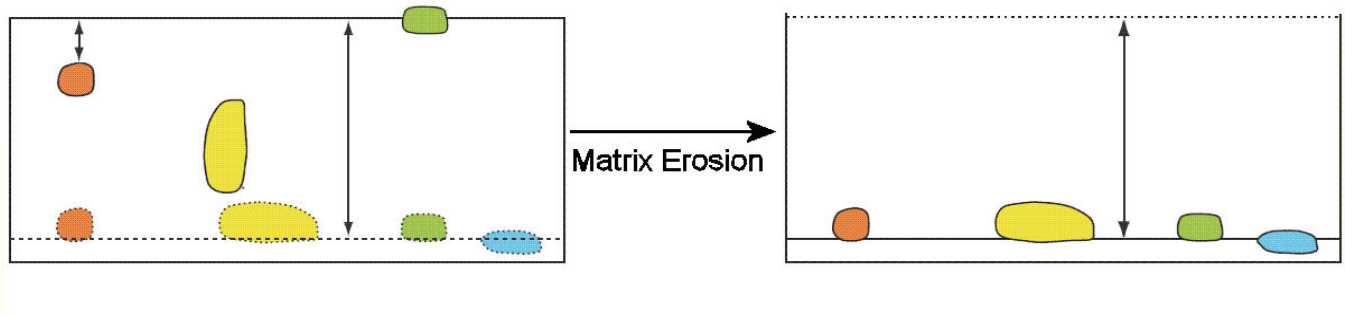

Figure 4: Matrix erosion gradually exposes previously buried boulders in a moraine. The exposure ages of all but the green boulder in this diagram would be younger than the true age of the moraine. Since the boulders' depths are evenly distributed, their apparent ages will be evenly distributed as well.

there is a gap between two groups of samples and the older group is much smaller than the younger group. The ideal model of inheritance makes no mention of any such gap, implying a significant failure of the model.

Two distinct kinds of erosion exist. On a hard surface like exposed bedrock or a boulder, the top surface of the rock will wear away gradually. Sometimes this wear is uneven due to weather conditions and the exact composition of the rock. On moraines, there is a different kind of erosion called "matrix erosion." Moraines are composed of a soil matrix with embedded boulders. Since the soil matrix is relatively soft, it erodes away much more quickly than the boulders, gradually exposing new boulders at the surface of the moraine, as in Figure 4. Despite the significant differences between these two types of erosion, experts have told me that only one mathematical model exists to describe both of them.

A final, even more worrying example is that, although vegetation cover is believed to affect apparent ages, no good model exists for this process. Perhaps because of this, experts prefer not to sample from areas covered by vegetation, but sometimes this is not possible; either way, this strategy does not account for areas that were once covered by vegetation but are not covered now.

Although experts are constantly working to improve their ability to explain and model various geologic processes, progress is slow and sporadic. Given these issues, it is obviously difficult even for human experts to select the best possible (or sometimes any) explanation for the spread in apparent sample ages using these mathematical models.

3.2. Heuristic Reasoning

Clearly there are major gaps in available knowledge. However, despite the incompleteness of mathematical models, isotope dating experts routinely reason about the modeled processes.

Because the models involved are known to be problematic, they cannot be relied upon exclusively. Instead, experts typically use these models to provide corroboration for the belief that a particular process explains their data, rather than as a method for choosing which process has affected the data. Despite the existence of mathematical models, then, Calvin will be unable to rely on these equations to provide trustworthy results. This is unusual in a scientific field; generally when a model exists it is accurate enough to provide useful results, and the knowledge embedded in the model can provide insight into the process being modelled. The incompleteness of mathematical models in the field of isotope dating makes reproducing expert reasoning an especially difficult process.

In part because of the gaps in formal understanding, some explanations are chosen by experts as a matter of convenience, rather than because of some objective value of the explanation. As in many fields, outliers are often removed from a dataset simply because they appear, statistically, to be outliers. Although experts prefer to find some valid explanation for the existence of the outlier (for example, a theory that it came from a nearby landform and was moved), no explanation seems to actually be required. Similar factors apply to the “lab error” explanation, which is generally used when no other explanation for a complete sample set seems reasonable. However, these poorly supported explanations are only used when no other explanation seems to apply, implying some general ordering on explanations that should be assigned to a dataset.

Because few samples are available to match the imprecise models, deciding what (non-convenience) explanation applies to a particular sample set is primarily done using a variety of heuristics. Some of these heuristics are based on a variety of knowledge about how particular types of landforms form and how each geological process affects different landforms (such as the difference between “regular” erosion and matrix erosion). Other heuristics concern distribution shapes and are applied to the apparent distribution of the small number of samples. These heuristics are not consistent: for instance, a set of three samples that are approximately evenly spaced will cause an expert to identify erosion as the most likely process affecting the sample distribution—not inheritance, as one would expect based on the *a priori* knowledge of sample distributions. This heuristic directly contradicts the “ideal” model of inheritance. It is apparent, then, that the knowledge base for cosmogenic isotope dating includes both vague knowledge and internal contradictions.

3.3. Solution Requirements

As discussed in Sections 3.1 and 3.2, many processes have similar manifestations in the data. This is true not only in the (imprecise, incomplete) mathematical models, but also in the heuristics used to diagnose various problems. Therefore, there is generally a fair amount of evidence both for and against several of the processes under consideration for explaining the data, which is a major challenge for automatic analysis. From my observation of experts in this field, it appears that a standard approach to this problem is to select one process and look for evidence both for and against that process. If it is possible to gather enough evidence in favor of a process and not possible to gather a similar or greater amount of evidence against the process, then it is considered a good candidate for explaining the data.

Any system that automates this reasoning must gracefully handle all of these problems. It will need to cope with imprecise, noisy, and incorrect data. Since mathematical models are generally incomplete or nonexistent, it must be capable of reasoning with heuristics applied to small amounts of data. Given the nature of the existing heuristics, an automated system will have to deal with uncertain heuristics that may contradict each other or support several different conclusions. This is a significant artificial intelligence challenge: most AI architectures are poorly suited to handling contradiction and uncertainty in their knowledge base[4].

In addition to these technical issues, experts are unlikely to agree with any conclusions made by Calvin unless they understand the reasoning behind those conclusions. Thus it is also critical to the usefulness of Calvin that it be capable of convincingly presenting the reasons for its conclusions. This capability will provide the additional benefit that students of geology can examine the reasoning and heuristics that are used in selecting a process. Although presenting a system's reasoning in a useful, human-readable form is not a new task for AI, this requirement does limit the types of reasoning appropriate to use in addressing the problem.

Although a perfect system would also be able to perform the leaps of insight involved in arriving at completely new explanations, this feature is a tall order for automatic reasoning and not a goal for Calvin. However, most of these leaps of insight involve the recognition of particular patterns in the data, and it is my intention for Calvin to *assist in* the recognition of these patterns. For example, the insight in (Zreda, Phillips, and Elmore 1994) required recognizing that a group of samples that did not fit the expected distribution were all taken from

4.Machine learning and other statistical systems might allow me to bypass these problems, but are unsuitable for other reasons: see Section 4.

boulders less than a particular size. Calvin may be able to find patterns like this, e.g. by using clustering techniques to identify what might be unusual about a problematic group of samples.

4. Rejected Approaches

4.1. Machine Learning

Although machine learning would allow me to solve the problems of building an automated system for cosmogenic isotope dating analysis quickly and without having to work so hard, machine learning requires training data. Lots of data. As discussed in Section 3, cosmogenic isotope dating is a comparatively new field and relatively little analysis data is available for learning. Generation of new data in this field is also comparatively slow. For example, during his more than twenty-year career in this field, Marek Zreda has dated only around a hundred landforms. Because of the scarcity of data, a knowledge-based system is a much better idea (Gaines 1989).

The other important reason not to apply machine learning to cosmogenic isotope dating is that few machine learning architectures are designed to present the reasoning behind their results to the user. Instead, they simply present an answer based on internal black magic. My experience with experts in this field indicates that no system for automating the analysis process will be acceptable unless experts can look under the hood and understand why the system has arrived at some particular conclusion. Therefore it seems clear that machine learning is not an acceptable architecture for Calvin.

4.2. Fully Symbolic Logics

Given that Calvin should be implemented using some sort of knowledge-based architecture, I considered a strictly symbolic system, such as first-order logic. Unfortunately, fully symbolic logics are notoriously bad at handling contradictions and partial support. Contradiction is especially troublesome, since in first-order logic any conclusion may be proven if there is a contradiction in the set of premises (rules). Some architectures do attempt to handle these weaknesses, but they are unfortunately insufficient for the purposes of the Calvin system. The symbolic architectures I have considered and rejected for Calvin include simple kinds of defeasible reasoning, truth maintenance systems (TMSs), and causal reasoning (e.g., counterfactuals).

Defeasible reasoning is a symbolic logic based on the premise that contradiction does not imply any fact about the world to be true. Instead, when a defeasible reasoning system proves a contradiction, it returns to the contradictory proofs. The weaker of the two proofs is defeated, and the system removes that proof from its knowledge about the world. Most systems of defeasible reasoning use systems of absolute proof and absolute defeat (Lucas 1997). However, the experts I am working with appear to reason using a system of partial support and defeat: one example of this is that they tend to demand less evidence for a conclusion that a moraine is eroding than they need to conclude that a lava flow is eroding. It is simple and intuitive, in view of this, to model moraines as partially supporting erosion and lava flows as partially defeating it.

A TMS uses a set of assumptions about what properties of the world are in force or not in force to handle contradiction. When the system arrives at some contradiction, it examines the assumptions that led to the proofs on both sides of the contradiction, alters the assumption set, and tries again. Some TMS architectures are able to model partial support and defeat in a probabilistic way (Forbus and De Kleer 1993). To use this architecture for Calvin, the TMS would need to contain one assumption for every possible rule in the system. The system would then determine all the subsets of rules that do not result in a contradiction, and the correct conclusion would be the one with the largest number of possible worlds supporting it (or possibly the largest probability sum over the possible worlds). Although this architecture might work, it seems convoluted and unintuitive. Furthermore, it does not appear to be an accurate model of how experts think: like most people, they appear to reason in terms of better or worse *evidence* for some conclusion, rather than according to sets of *possible worlds*. Presenting the reasoning of such an architecture in a comprehensible way also appears to be a significant challenge, since the system would somehow have to explain the reasoning about all the many possible worlds in a coherent and manageable way.

A final symbolic tactic to consider is an architecture focused around causal reasoning. This approach would still need to handle issues of partial support, but the amount of explicit contradiction might be less. The specific method of causal reasoning I considered is counterfactuals, which convert the causal assertion “A caused B” to the statement “if it were not for A, then B would not be so” (Pearl 1999). Since it is my goal to not only “solve” the problem of isotope dating, but to do so in a way that accurately models expert reasoning, this model is inappropriate. Experts approach their task from a diagnostic perspective, reasoning backwards from sample data rather than forwards from geological processes. Furthermore, a causal

architecture suffers even more from the incomplete understanding of various processes: it is difficult to state the result of, e.g., cover by vegetation, since experts have no definitive model of how it works.

4.3. Bayesian Networks

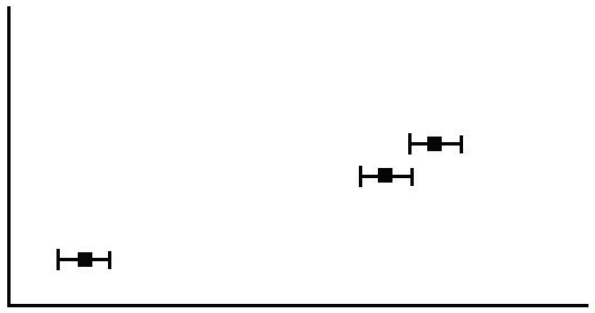

Figure 5: A possible set of samples from a single landform. Using pseudo-statistical reasoning, experts prefer not to choose the left most sample as the age of the landform, because the other two are so distant from it and close to each other.

Bayesian networks are often presented as solutions for various problems in artificial intelligence, particularly the problems of partial support and contradiction (Pearl 1988). However, Bayesian networks are aimed specifically at problems where multiple prior and conditional probabilities are known and defined. In the case of cosmogenic isotope dating, I do not know these various probabilities. Rather, I have general heuristics and vague knowledge about weights. In fact, it seems unreasonable to represent certain expert tendencies as probabilities. For instance, some explanations of data lead to a choice of a single sample, youngest or oldest, as representative of the correct age of the landform (all other samples are discarded). However, given a sample distribution like the one in Figure 5, experts are reluctant to choose an explanation that requires them to select the youngest sample because it is so distant from the other samples, which are near each other. Although this pseudo-statistical reasoning could be represented probabilistically, doing so is unintuitive because the reasoning is inherently not probabilistic. Finally, my attempt to organize the rules in the prototype version of Calvin into a Bayesian network led to a network with several cycles and some nodes with many paths in or out. A network representing all the knowledge of isotope dating experts would likely be difficult to maintain and computationally complex to use. For all of these reasons, it seems that Bayesian networks are not an appropriate solution for Calvin.

4.4. Schemas/Case-based Reasoning

One final architecture to consider for Calvin is a form of case-based reasoning (Kolodner 1993), (Cunningham, Bonzano, and Smyth 1995), and (Clark 1989). The reasoning from a previously solved case is presented, largely intact, as applicable to the case under consideration. Like machine learning, pure case-based reasoning requires a large number of

fully-analyzed cases so that any new case presented to the system resembles some previous case closely enough to draw conclusions based on known cases. Modifications to case-based reasoning exist that attempt to allow its use in situations where an insufficient number of previous cases exist (Surma and Vanhoof 1995), (Turner 1992).

(Turner 1992) uses abstracted cases called schemas to perform diagnosis. In his system, a number of default or prototypical cases are defined that cover any normal problems that the system is likely to encounter. When a symptom is unique to a particular type of disease, the system considers diagnosing that disease. If the symptoms expected for that disease are observed, then it is considered a correct diagnosis. Abnormalities are defined by hand, usually by writing exceptions to the expected symptoms. His system's results can be presented to the user as the text written by experts who analyzed the original schemas, along with any exceptions used by the system for that case.

Although a schema-based system would solve many of the problems important to Calvin, I do not feel it is an adequate solution. In particular, the analyses shown to me by experts include so many exceptions to any "normal" case that it seems impossible to define what cases might be normal in this field. Since every new problem is likely to contain multiple exceptions to any schemas I could define, most of the knowledge used by the system would be in individual, hand-written exceptions. I feel it makes more sense to instead use an architecture that involves only individual rules, removing the need to identify "normal" situations in addition to the individualistic exceptions.

5. Intuitive Solution: Argumentation

Argumentation is a good solution to the challenges of automating cosmogenic isotope dating. Argumentation systems are symbolic, so I need experts (which I have) instead of data (which I don't have) to write a complete system. Because it uses symbolic logic, an argumentation system can present the reasoning behind its conclusions in a clear and comprehensible way. Reasoning in an argumentation system can use weights, allowing the system to elegantly represent partial support. In addition, argumentation systems are explicitly designed around the idea of contradiction, so contradictory solutions do not present them with any challenge (Krause, Ambler, Elvang-Gøransson, and Fox 1995) (Doyle 1983). Unlike in a Bayesian system, the weight associated with any particular piece of evidence is not understood as a probability, so there is no need to extract probabilities from vague heuristics. Argumentation

does not require me to define what is “normal” and what is “exceptional” in any particular case, only what pieces of evidence are more compelling, so it does not have the failings of a schema-based system. Finally, the structure and reasoning of an argumentation system appears to closely match the reasoning of experts in the field, making it an intuitive and easily understood architecture for the problem at hand.

Several different logical systems have been labelled “argumentation logics” by various authors, for instance (Dung 1995), (Farley 1997), (Prakken 1996), and (Vreeswijk 1991 and 1993). These systems are discussed in Section 6.2. In general, the term is applied to a symbolic logic built explicitly around the idea of contradiction among rules and/or evidence. All argumentation systems contain the notion of an argument, akin to a proof in first-order logic, along with some mechanism for defeat among arguments. Argumentation systems may also contain mechanisms for defeating rules and input data. Most argumentation systems deal in absolute support and defeat: if an argument exists for a conclusion the conclusion is believed, until the argument is defeated. Typically, if an argument is defeated, it is defeated absolutely, but some architectures do contain mechanisms for both partial support and partial defeat. Defeat in argumentation systems typically comes in two flavors: undercutting and rebuttal. Undercutting refers to the case in which the premises one argument is based on are attacked (contradicted) by another argument. Rebuttal is the direct attack of one argument on the conclusions of another.

The argumentation framework used by Calvin is based on the Logic of Argumentation (LA) introduced by Krause et al. (Krause, Ambler, Elvang-Gøransson, and Fox 1995), which defines an argumentation logic that allows for both partial support of conclusions and partial defeat of arguments. Rules and input data are not explicitly defeasible in this system. It is implemented as a labelled logic: rules and data are labelled with a confidence level that is used to determine which of two arguments is stronger. It is important to note that these confidence levels need not be continuous or numeric—Krause et al. use the set {*certain*, *confirmed*, *probable*, *plausible*, *supported*, *open*}, with optional negative terms such as *opposed* or *doubted* as an example set of confidence values. In this case, the confidence in an argument is related to whether it is internally consistent and whether undercutting or rebutting arguments for it also exist. As arguments are built from rules and input data, confidences propagate to their conclusions using some system of combination. The labelled logic allows the definition of an appropriate set of confidence values and combinations for each individual application. Conclusions in the system are labelled with the arguments for and against them, so that as new information is discovered,

the arguments about a conclusion can be examined and possibly defeated, possibly leading to an overall defeat of the conclusion they supported.

LA uses a version of first order logic allowing only the And and Implication operators (Or and Not are not used). Calvin uses this same extremely simple logical system, so all rules are expressed as a conjunction of literals implying some conclusion. When Calvin generates conclusions, it labels them with the rules and input data used to create them. I have also implemented a Not operator for Calvin by using negative confidence values. In contrast to LA, however, Calvin does not currently use an explicit notion of confidence for rules, only constructed arguments and conclusions have actual confidence values. Rules contain information about how to translate confidence in their premises into confidence in their conclusions rather than a direct confidence value. Finally, Krause et al. do not give a mechanism for combining confidence in several arguments into overall confidence in a conclusion that is well suited to the problem solved by Calvin.

A good metaphor for the way the prototype version of Calvin treats arguments for and against conclusions under consideration is the idea of grains of sand on a scale. Every argument for or against a particular conclusion, no matter how weak, adds some weight to its side of the scale. A large enough number of very weak arguments can overpower and defeat a smaller number of stronger arguments. Intuitively, an argument is stronger if experts will find it more believable. I have chosen this metric over some more objective measure such as truth or likelihood because the field of isotope dating is so new; few absolute truths exist, and it makes more sense to rely on expert judgement. This means that arguments formed from rules and input data with higher confidence values should be stronger than data and rules in which the system has less confidence. In a break from the scale metaphor, the strengths of arguments added to one side are not precisely additive. Instead, Calvin uses a system of diminishing returns so that every additional argument of the same independent strength adds less absolute weight to its side of the conclusion. The idea here is that every additional argument for or against a conclusion adds its confidence to the confidence “unclaimed” by other arguments. Given two arguments for the same conclusion of strength 0.4, the total confidence in the conclusion is 0.4 + 0.4 * (1 - 0.4) = 0.64. Thus two arguments of strength 0.4 are not as strong as one argument of strength 0.7, although three 0.4 arguments would be stronger than one 0.7 argument. The goal here is to allow weaker arguments to eventually overpower stronger ones while being mindful of the fact that stronger arguments are stronger for a reason. (It should take a great deal of minor nitpicking around the edges to overcome one very strong argument!)

Calvin currently uses a confidence set of continuous values in the range [-1, 1]. Rather than using explicit negation of conclusions, negative values imply confidence in the negation of the conclusion to which they refer. Thus a confidence value of -1 implies absolute confidence (certainty) in the negation of some literal, and a confidence value of 1 indicates absolute confidence in the literal to which it is attached. A confidence value of 0 means that the system has no knowledge whatsoever about some literal, and thus has no opinion either way. It is not my intention to present these numeric confidence values to the user in the final form of Calvin lest they be interpreted as probabilities (or otherwise independently significant numbers). In addition, since these numbers have essentially no significance except in relation to each other, they are somewhat difficult to enter into the system initially. There is some evidence that precisely what numbers are chosen has little effect on system behavior (Doyle 1983). A better method for handling confidence values and combinations is a major future milestone for this work. It is likely that a symbolic confidence mechanism will be able to give the same results as the current system in a less contrived way. More detail on possible solutions is discussed in Section 9 with my plan of work.

6. Related Work

Many diagnostic systems solve problems similar to the one addressed by Calvin, in which there is some normal, expected behavior and the causes of divergences from this behavior must be diagnosed. In isotope dating, the expected pattern is that all samples have the same apparent age, and the task is to diagnose what process has caused a spread in apparent ages. The predominant paradigm in medical diagnosis is to build a complete model of a system and to use that model to make predictions about what symptoms are expected from what diseases (Lucas 1997), (Struss 2004), and (Santos 1991). (Santos 1991) uses a model-based approach that can present multiple possible diagnoses generated by trying different assumption sets within the model. A model-based methodology is not suited to my particular domain because complete models of most geologic processes simply do not exist. In addition, model-based systems are not as well suited to handling contradiction, especially if it occurs within the model. It is difficult for a model to predict a set of symptoms from a disease when some symptoms are predicted to both occur and not occur!

Diagnosis systems that handle contradiction do exist, for example (Doyle 1983) and (Gaines 1996). (Doyle 1983) uses a rule scheme with default rules and explicitly defines under what

situations the default rules do not apply. (Gaines 1996) is designed around the idea of representing knowledge as trees instead of rules. The root node of the tree is the default situation or diagnosis. When a parent node is true (the condition in the node is met), child nodes are explored to see if their conditions are also met. Some nodes will then give an output, but no output is reported to the user until all allowable paths have been explored.

Even the diagnosis architectures that handle contradiction use “absolute” rules—it is not possible to express the idea that some data may only partially support a conclusion, or that a previously found conclusion may be only partially defeated. Instead, the conditions under which a rule does *not* provide support are explicitly encoded in the system, and rules must be absolutely defeated or not defeated at all. Calvin, in contrast, must have rules that partially support conclusions in order to accurately reflect the reasoning process used by experts. For instance, experts are more likely to accept (and require more evidence to reject) an “erosion” conclusion for a moraine than for other landform types. Trying to model this behavior without an ability to express partial support would be extremely difficult. On the other hand, all of these diagnosis systems are capable of presenting their reasoning to the user to help convince experts of initially rejected conclusions, an important feature of Calvin.

Several authors have discussed the virtues of presenting the reasoning behind a system’s conclusions in the form of trees or graphs, including (Bouwer and Bredeweg 2002) and (Gaines 1996). (Bouwer and Bredeweg 2002) observe that a system’s complex results are much more easily understood in a structured, visual format such as a state transition graph. (Gaines 1996) explicitly represents the entire rule set of a system as a set of trees. The system can present answers in terms of these trees, leading to more-comprehensible results. Other characteristics of a system’s presentation of results are also important. (Boy and Gruber 1990) and others discuss the importance of keeping a human “in the loop.” They argue that systems that automate some but not all of the tasks performed by a user are more useful than systems that attempt to automate all tasks. In addition, such systems are much more likely to enhance a user’s learning. (Puyol-Gruart, Godo, and Sierra 1992) points out that even when a particular conclusion cannot be reached by a reasoning system, it is likely that presenting what the system has managed to determine will be useful to the user. Although the current output of Calvin is text-based, it is my intention to eventually present the system’s reasoning to the user in a tree-based format to mirror the arguments’ internal structure. My plan for the user interface is discussed in detail in Section 9.

6.1. Various nonmonotonic logics

Several kinds of defeasible reasoning besides argumentation have been put forth by various authors. These include circumscription (McCarthy 1980), (McCarthy 1986), default reasoning (Reiter 1980), (Doyle 1983), and other forms of nonmonotonic reasoning (Pereira, Alferes, and Apar'icio 1991), (Gaines 1996). While each of these is useful for many problems, none of them solves all of the issues involved in cosmogenic isotope dating analysis. Furthermore, many nonmonotonic logics are unsound in some way, making their use problematic (Etherington, Kraus, and Perlis 1991).

Circumscription allows the definition of normal situations and all the cases in which they might be circumscribed. It requires the definition of specific aspects that are abnormal only in abnormal situations, so that it is necessary to create a large number of "aspect" variables to express all of the possible abnormal situations. It only allows for complete defeat of any particular rule—a rule is defeated and replaced by some other rule precisely when one of the abnormal situations holds, and never otherwise.

Default reasoning uses rules with default conclusions and then defines specific exceptions where they do not apply. This is similar to circumscription in that every situation in which a rule does not apply must be defined, and in that rules are always defeated absolutely. The primary difference between default reasoning and circumscription is that default reasoning does not require that a rule be replaced by another when it is defeated; the rule is simply no longer available for reasoning about the current case. The guards on Calvin rules are reminiscent of this method of defeat, since the guards define cases when a rule is not applicable and should be functionally removed from the database for the current problem.

The nonmonotonic logic defined in (Pereira, Alferes, and Apar'icio 1991) allows exceptions to rules in a number of ways. Rules may be absolutely, normally, or possibly true in the rule database. Rules that are normally true can be defined with exceptions either to the conclusion of the rule or to the rule itself. Rules that are possibly true allow their conclusion to be drawn or not drawn as the situation appears to dictate (generally the choice that avoids a contradiction in reasoning is used). Although several kinds of defeat are allowed (including the defeat of exceptions to rules), defeat is once again absolute and cases where rules do not apply are generally explicitly defined. Possibly-true rules do not require the definition of specific exceptional cases, but a system written with only possible rules may find it difficult to reach any conclusions at all.

As discussed above, (Gaines 1996) uses a tree structure for rules with default conclusions at the root and repeated refinements or rejections of the initial conclusion(s) as the tree is descended. The resulting logic is similar to the other nonmonotonic logics discussed above: exceptions to rules are explicitly defined and defeat is generally absolute. The primary distinction is in the visual structure of the rules rather than a functional difference in logic.

(Pollock 1994) also uses a visual, graph-based system of logic. A reasoning graph is composed of nodes representing literals and two kinds of directed edges. These edges capture cases where the literal of one node is needed to conclude the literal of the other and where the literal of one node forbids (defeats) the literal of the other. If every consistent status assignment (defaulting to undefeated) fails to defeat a node, then that literal is accepted in the final result.

The two common themes among these various defeasible reasoning systems are a need to explicitly define situations when rules do not apply and an absolute defeat system for rules, conclusions, or both. Explicit definition of exceptional situations is beneficial from a purely theoretical standpoint because it allows experts to define the situations in which they know that a rule does not apply, rather than having to make up a number out of thin air for every situation. However, when the knowledge base in a field is still enormously incomplete, experts are less likely to know of all the possible exceptions, and simply picking a confidence number (or simple symbolic value) is a more practical choice when defining rules that do not always apply. Absolute defeat of a rule or conclusion is also problematic for Calvin. As a practical example, I have observed that experts are unlikely to prefer an erosion conclusion on lava flows. Therefore I believe that the fact that the landform under consideration is a lava flow is somehow evidence against erosion. On the other hand, erosion on a lava flow is certainly possible. I need a way to represent the fact that “landform = lava flow” is a small piece of evidence against erosion, but does not absolutely defeat erosion as a conclusion. The logics discussed above are unsuited to a reasoning process built around this kind of supporting and detracting evidence.

6.2. Argumentation

There appears to be no consistent definition of the term argumentation. It may be applied to systems that aid in human discussion (Brashers, Adkins, Meyers, and Mittleman 1995), systems that arbitrate between agents (Parsons, Sierra, and Jennings 1998), or systems that do automatic problem solving like Calvin. Within the field of argumentation as an automatic logic system, an impressive variety of subtly different logics have been defined. Most of their differences revolve around the question of when an argument should be defeated. As with the

nonmonotonic logics discussed in Section 6.1, most argumentation systems require a single rule to absolutely defeat a rule, rather than allowing a set of rules to collectively overwhelm another rule or conclusion. Also variable is whether argumentation is viewed from a dialectical standpoint or as an extension of first-order logic.

(Dung 1995) takes a dialectical approach to the problem of argumentation. Arguments in his system are defeated on the basis of which side "has the last word." In other words, one party in the dialog presents a premise. It is then the goal of the other party to attack that premise with some other argument, defeating it absolutely unless the new argument can be defeated. The original arguer then attacks the attack, and so on until one party has no available attacks. When one chain of argument is finished, the losing side has the option to branch out from some state further up the tree and make a second attacking argument against a previous argument by the opponent. The side with the last standing argument is the winner, with the conclusion accepted or defeated appropriately.

(Prakken 1996) also casts argumentation as a dialectical process involving two parties. Here it is the case that the conclusion under consideration is by default rejected. Thus it is the goal of the proponent party to prove the conclusion, and the goal of the opponent to prevent the acceptance of the proponent's arguments. The burden of proof is on the side of the proponent, allowing the opponent to win simply by finding a single attack on the proponent's argument that cannot be defeated. Arguments are explicitly based on assumptions and rules. An argument can be attacked by either via its assumption or via the rule on which it is based.

The logical argumentation system in (Vreeswijk 1991 and 1993) is based on the idea that when arguments exist both for and against a conclusion, the strongest single argument should win. Arguments come in two types: statistically based and generically based. Statistically based arguments will always defeat any opposing generically based arguments. However, among a set of generic arguments, the author argues that no reasonable system of defeat can be defined. Rather, a set of mutually contradictory generic arguments allow a split into two equally believable possible worlds. This situation comes about because all generic arguments are taken to have precisely equal force. In Section 4.2 I discussed some problems with generating sets of possible worlds; these problems are even more pronounced since the system in (Vreeswijk 1991 and 1993) does not even give a relative ranking of the possible worlds it has generated.

(Farley 1997) uses argumentation as a method for performing qualitative simulation in the presence of conflicting indications. His system allows three different modes: accept all

arguments, accept the side with more arguments, and accept all defeaters. In the first mode, no conclusion can be defeated. Every conclusion is accepted if there is any undefeated argument for it. This mode allows a user to explore all the possibilities of a system and might guide experimentation. In the second mode, all arguments have the same strength, and the side with more undefeated arguments "wins." Finally, in the third mode, a conclusion is accepted if and only if there is an undefeated argument for it, and all arguments against it are defeated. This is reminiscent of the requirement in law that a person must be proven guilty beyond all reasonable doubt. Three classes of arguments are defined, with a hierarchical defeat mechanism between them. Arguments are explicitly defeated only when they are attacked by an argument with a higher type in the hierarchy; cases with arguments of the same strength are handled by the system modes.

All of the above systems use some form of absolute defeat of an argument. Again, this mechanism is unsuitable for my problem for the reasons discussed in Section 6.1. However, Calvin does resemble several of these systems in various ways. For example, like the system presented in (Vreeswijk 1991 and 1993), Calvin chooses to present all arguments to the user rather than declaring an argument defeated and ignoring it. In contrast to the approach in that paper, however, Calvin arguments are presented with some measure of how strong each argument is, so that the user does not need to filter among a large number of arguments. The hierarchy of rules defined in (Prakken 1996) has much the same effect as the confidence system currently being used in Calvin, especially in rules with a scalar-type confidence combination. Rules with larger absolute confidence values are implicitly higher in the Calvin hierarchy, since rules with larger scalars will defeat those with smaller scalars, presuming they are using knowledge with the same confidence values. (Prakken 1996) does not take into account the possibility of better knowledge being used in a rule, as Calvin does. The second acceptance mode of (Farley 1997) bears an interesting resemblance to the fact that Calvin conclusions gain more confidence as more arguments for them are constructed. On the other hand, (Farley 1997)'s arguments all have the same weight, whereas Calvin arguments can have a range of weights, allowing the system to differentiate between the quality of arguments.

It should be noted that all of the papers referenced in this section simply define systems of logical argumentation. Although some of these papers give small examples showing how their logic can be used, few results exist for applying argumentation to large problems. Calvin will be a large-scale, practical analysis system built using argumentation, not another floating system of argumentation logic without an implementation.

7. System Design

7.1. The Engine

Calvin takes as input all the data the expert has collected about a set of samples. This includes the chemical composition data from the AMS lab, the apparent ages of the samples (which are calculated by ACE), and all the observational data from the sample site. This includes both qualitative and quantitative data. The qualitative data includes observations such as the type of landform where the samples were collected and the color of the sampled boulders. Quantitative data includes information such as latitude and longitude, the elevation of each sample, the calculated apparent sample ages, etc. Although it is not part of the current implementation, future implementations of Calvin may also request additional data about the sample site, such as the average annual temperature or snowfall. Some of this extra information may be costly or difficult to obtain, so Calvin should request it only if it is likely to improve the system's results. Since single landforms are not necessarily dated in isolation, input may also include information about nearby landforms.

Calvin considers a list of processes that may have affected the landform. Currently the engine assumes that any process may affect any landform; this assumption may change as more knowledge is added to the system. Processes that are more common (according to experts), such as erosion and inheritance, are considered before less-common processes like snow cover. The current Calvin engine considers processes in groups, and only goes on to the processes in a lower group if the arguments for the higher groups are considered insufficiently convincing, based on the confidence in each process. The top group contains the possible conclusions "no process," "erosion," and "inheritance," all of which experts tell me must always be considered for every set of samples. The other processes the engine may argue about are "snow cover" and "outlier," respectively, in two separate groups. Since the current implementation of Calvin is a prototype, this list is not yet complete. As more processes are added, the "outlier" explanation will remain in its own group so that it is always considered last, and only if all other explanations have failed. This is intended to prevent Calvin from overusing the outlier process.

When a sufficiently convincing argument is built for some process, the engine does not move on to the next group (although it will finish building arguments for the current group). Instead, the engine presents its completed arguments to the user. Users can choose to have the system

generate more arguments if this set is deemed insufficient. This feature is predicated on the assumption that Calvin will occasionally be wrong, and expect the user to be convinced by an argument that is invalid in some way[5]. When the user requests that additional arguments be built, the engine simply moves down the priority list from its previous halting point, and presents the arguments for each process in the next group. The user is then given the option to finish or continue building additional arguments.

The Calvin engine builds arguments in a top-down manner, working from the conclusion (such as a particular process) under consideration. Rules that apply to this conclusion are found in the rule database (just a list of rules; described in Section 7.3), and unification is applied to find the second level of conclusions to be considered. This process continues until the information required for unification is in the data initially input by the user. If the required data is not in the inputs, the engine assumes that the rule or rules under consideration are not applicable in this case. A list of these rules might later be used to request additional data from the user, although this feature is not implemented in the prototype. Currently, Calvin builds the most complete argument set possible from its rule database and input data for every conclusion. However, as more knowledge is added to the system I am alert to the possibility of performance problems, and I have features in place in the current system to allow Calvin to build less-complete arguments, avoiding the most computationally-intensive rules unless they are needed. This functionality is discussed in Section 7.4.

7.2. Arguments

Rules in Calvin are designed on the standard Horn clause structure of first-order logic. The primary portion of a Calvin rule is an implication of the form A=>C, where A may be a single literal or the conjunction of several literals. Since classical first-order logic systems are built around a set of rules written in the form of an implication, it makes sense that their general goal is to prove the premises of the implication true to prove the conclusion true. When they are able to do this, the conclusion is known to be absolutely true and can never be rescinded. Calvin works in an analogous fashion, except that it allows for the possible eventual defeat of any conclusion. When there are arguments for the antecedents of a rule, Calvin combines them with the rule to obtain an argument for the conclusion of the rule. However, Calvin does not believe

5.This option also supports users who are simply curious about what arguments exist for conclusions the system has not yet considered.

the conclusion with absolute confidence—stronger arguments against it (or against the conclusions used to form it) may be found, and the current argument overturned.

An argument for some conclusion C is a collection of trees built from these rules. The top node of each tree is a rule whose conclusion is C, such as the rule A=>C. The children of this node are each a complete argument for one of the literals in A. The leaves of the argument trees are taken directly from the data entered by the user. Each tree in the argument has a confidence, derived from the rules and input data in the entire tree. The confidence in the overall argument is a function of the confidences in the individual trees. As described in Section 5, “pro” arguments are placed on one side of a balance and “con” arguments on the other to see how convincing the overall argument is for the conclusion. When building arguments, the backwards-chaining engine makes no distinction among confidences. Thus a rule of the form A=>C may still be used in the arguments for C when A has a negative confidence value. However, because of the way the rules and confidence combinations are written, this situation will generally result in a tree that argues against C rather than for it. Many of the “con” trees in a typical argument are created in this way.

Figure 6 shows an example collection of argument trees in Calvin which might be translated into English as: “Erosion is a supported by the fact that moraines are likely to erode and this landform is a moraine. However, there is no visual evidence of erosion such as a flat crest or weathering, making erosion a less convincing conclusion.” The final system will present output in a tree format, making the structure of the arguments readily apparent. Arguments presented in English would be exciting, and I will implement them if time allows.

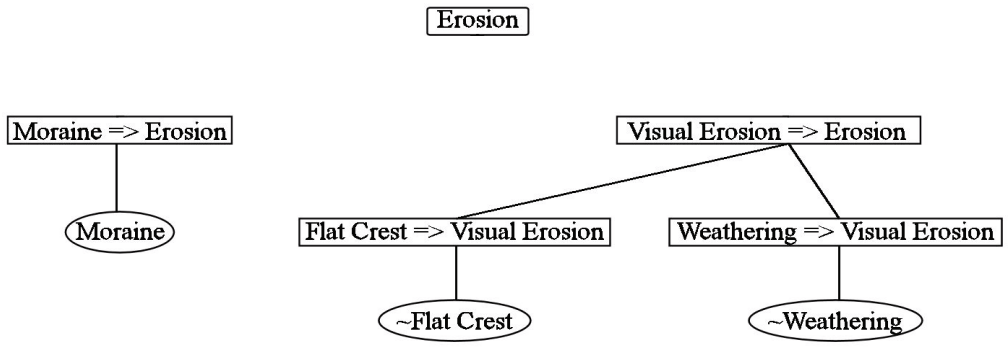

Figure 6: An example set of Calvin argument trees. At the top is the conclusion being argued about. Applied rules are in boxes; input data is in ovals. My prototype system contains more rules than are shown regarding erosion; these trees are a subset for illustration.

7.3. Confidence

Calvin's confidence values are 2-dimensional vectors containing a 'Match' and a 'Quality' value. Match values express how closely the actual input data matches the proposition in a rule. They are composed of two components, a direction and a degree. The degrees are ‘somewhat,’ ‘no modifier,’ and ‘very,’ and the directions are true and false. These degrees are used to handle noisy input values and vague knowledge. For example, consider the rule 'x > 10 => y.' It is not necesarily clear why the value 10 is important, rather than 9.9 or 10.1; for many real-world rules, such a threshold is only a guess. In addition, when x is noisy and its true value is close to 10, the observed value may fall on either side of the cutoff. As a result, x values near 10 result in 'somewhat' true or false matches. The further the observed value of x is from 10, the more extreme the match component of the system's confidence in y. In principle, Calvin could use more match levels, but these three have so far been sufficient in practice. This mechanism provides functionality similar to fuzzy logic.

The Quality component of a confidence value captures the fact that some rules are better than others. That is, some rules are derived from generally accepted theories, whereas others may be derived from an expert's experience with a single past case. Calvin uses four levels of quality: poor, okay, good, and definite. Two quality values are assigned to each rule based on experts' judgments of the quality of their own knowledge. One value is used if the premises in the rule’s implication are true, and the other if they are false. In the example, the quality dimension of the confidence in y is directly related to one of qualities of the rule

Confidence values are used in three different situations: when a chain of arguments is constructed for a conclusion, when two arguments apply to the same conclusion, and when comparing the confidence in different conclusions. The first situation occurs when Calvin is considering a proposition z, and knows that x > 10 => y and y => z. Calvin determines its confidence in y using the match value for x > 10 and the quality of the first rule. The confidence in z is the same match value and the lower quality of the second rule or the confidence in y, so that the argument for z is only as high-quality as the least trustworthy rule used in creating it.

Confidences are combined when multiple arguments can be built for the same conclusion. In this case, the confidence with the single highest quality is selected as the overall confidence. If multiple confidences have the same quality and the same match direction, the one with the most extreme match is chosen as the overall confidence. The most complicated case is when arguments with the same quality and opposite match directions exist. In this case, Calvin takes

a vote among arguments of the highest quality to determine its confidence in the conclusion. When the vote is a tie, arguments at the next lowest quality level are considered. An overall tie is considered slight evidence against the conclusion (because there is apparently no convincing evidence for it). Otherwise, the final confidence has a match value based on the margin of victory and quality level where the voting ended.

Finally, confidence levels are compared to present a final conclusion to the user. Higher-quality conclusions are considered better, with better matches breaking ties.

7.4. Rules

In addition to the implication that is the main part of the rule, a Calvin rule contains guards to keep it from being used when it is inappropriate, as well as instructions for how to combine the confidences in its antecedents when building an argument using the rule. Many rules contain elements that act as internal variables to pass data to the next conclusion or among the conclusions of the rule. These internal variables may hold temporary values, such as a mean sample age, or they may be parameters to the conclusion of the rule, for example a variable that identifies which sample is a possible outlier in a rule whose conclusion deals with an outlier.

The guards on Calvin rules prevent the system from building arguments using rules that are not applicable to the current case. An example of where this feature is needed is in the case of snow cover. Calvin has a rule that snow cover is more likely if calculated sample ages are younger at higher elevations. This is based on the knowledge that snow cover blocks cosmic rays and more snow falls at higher elevations. Obviously the rule only makes sense for sample sets with elevation ranges that are large enough to have different levels of snow cover at the opposite ends of the range. The guard on the rule enforces that this precondition always holds when the rule is used.

7.5. Evidence

The data Calvin uses to draw its conclusions is referred to as evidence. The antecedents in a rule's implication are patterns, or templates, for evidence—they indicate what evidence will be needed to satisfy the rule. These patterns tell the backwards-chaining engine where to look for information that will allow it to use the rule (whether to build an argument or look in the input data, for example). The actual data found by the engine is the evidence. The system has four different kinds of evidence (and therefore four types of patterns for evidence): observations, simple calculations, complex calculations, and arguments. The distinction between these types

of evidence is inspired by the PRET (Stolle and Bradley 1996) system. The separation allows less computationally intensive rules to be considered first.

7.5.1. Observations: Observations are direct uses of data that was entered by the user. Usually an observation is some binary quantity—for example, whether a sample has an apparent age less than a certain value, or whether the landform type is a moraine. Because the user's quantitative observations are generally assumed to be noisy, the system has more confidence in a given observation when the values in the relation are farther apart. This makes sense: when a noisy observation is quite close to some cutoff value, it is more likely that the actual relation to the cutoff is reversed (the noise has moved the observation across the cutoff). More concretely, for an antecedent like "elevation < 10000 ft.," the system will be much more confident that the condition has been met with an elevation value of 7000 ft. than a value of 9999 ft. Observations may also take the form of a quantifier such as for-all or there-exists. These are handled by selecting the highest (for there-exists) or lowest (for for-all) individual confidence value among the quantified entities. I am still considering how to handle the potential untrustworthiness of qualitative evidence, so Calvin currently has absolute confidence in qualitative considerations.

7.5.2. Simple Calculations: Simple calculations are generally calculations of simple statistical properties of entered data. They are used solely for the purpose of generating the calculation's results and all simple calculations have a confidence value of 1. A simple calculation might find the mean of all apparent sample ages and assign it to a variable that can be used in another part of the rule. This other part of the rule could then perform some other check using this data, such as ascertaining whether all apparent sample ages fall within a certain distance of this mean. Because all simple calculations have the same confidence value, their confidences are ignored in determining the confidence of the overall argument using the rule. The purpose of the confidence value is to make all types of evidence behave in a similar manner.

7.5.3. Simulations: More complex calculations are called "simulations" because many of the complex calculations used by experts involve simulating a possible situation to see how closely it matches the current data. The most common type of complex calculation implements a simulation of this variety. Other types of complex calculations in Calvin include checking for correlations between two fields and analyzing the similarity between two different samples' chemistries.

Unlike simple calculations, simulations have varying confidence values based on their results. For example, when checking for a trend between elevation and age, the confidence of the simulation will be higher when the trend is stronger. Some simulations may also provide a return value for use by other parts of the rule. For instance, a simulation that determines the annual amount of erosion most likely to produce the actual sample distribution would return the calculated erosion level so that the feasibility of that erosion level could be checked.

Simulations are implemented as procedures called by the engine as it examines the rules, allowing them to be as complex as necessary. As an example of this complexity, one simulation examining inheritance removes older samples from the input data and then calls the engine to build arguments about this changed input data. When only a few of the oldest samples need to be removed for the engine to build a good argument for “no process,” inheritance affecting only a few samples is a likely conclusion. One drawback of this mechanism is that, although Calvin is designed to allow the easy addition of new rules as new methods are found for reasoning about isotope dating, adding new simulations will require actually writing the code that performs the simulation.

7.5.4. Arguments: Sometimes the antecedents of a rule cannot be directly gleaned from the input data. In this case it is necessary to build a sub-argument for an antecedent and to use the sub-argument as evidence. For example, snow cover is much less likely in areas that are not cold. Calvin can build a sub-argument for whether the sampling area is cold as part of an overall argument about snow cover. As with the top-level argument about snow cover, the argument about whether the sample site is cold is currently built to be as complete as possible. Once again, this may change as more knowledge is added and performance becomes an issue.

8. Calvin in Action

The prototype version of Calvin incorporates only a small subset of the existing body of expert knowledge in isotope dating. Specifically, it contains rules that reason with sample ages, error quantities, location, boulder size, and elevation. Experts record and reason with many more observations about samples, including their precise chemistry, angle of the sampled surface, where exactly from the surface the sample was taken, shielding from other landforms, and exact sample dimensions. Sample chemistry seems to be especially useful in some cases for determining what process has affected a given landform. In addition to information about specific samples, experts record (and Calvin takes as input) general information about the

landform being sampled, such as the landform type, its location in relation to other landforms, and whether it shows visual signs of erosion (e.g., a flat moraine crest). It is my aim to incorporate as many of these expert observations and reasoning tools as possible into Calvin.

Because it does not yet contain rules for much of the data experts have about samples, the prototype version of the system is only able to reason about comparatively simplistic cases. Most of the system's current rules deal with the distribution of sample ages, size of errors, and trends relating these properties to other observations. This is not an unreasonable first step, since experts usually present this information first and argue primarily about these properties when demonstrating to me what process has affected a given landform. Test cases handling only these observations are nonetheless quite simplistic compared to real-world cases, which have more and richer data to guide reasoning and conclusions. The general reasoning methods of experts appear to be the same regardless of the level of detail they are presented with about a dataset. The same should be true of the Calvin architecture.

As a baseline, I constructed several simple test cases to be ideal versions of the ages and errors expected in datasets affected by a specific process. For each test case I selected a process and designed a sample distribution that closely matched examples I had seen of the results of that process. I then added additional information that had sometimes surfaced when discussing real-world cases with experts, such as visual indications of erosion or distant sample locations. These test cases address some of the challenges of real-world problems in the field (small number of samples, unimportant data included) while ignoring others (little noise, support for correct process is strong).

To verify my constructed test cases, I presented them as a visual plot of the sample age distribution to one of the experts I am working with, Chris Zweck, and asked him to identify what process seemed most likely, based on the small amount of data available. In each case he identified the process I had constructed the test case around, despite the small amount of information available. Unsurprisingly, his reasoning was fairly simplistic since the test cases themselves are fairly simplistic. Chris reached the same conclusion about the process affecting the sample set when presented with the additional information in the test case (rather than only the sample age distribution), but he seemed more confident in his conclusions when he could also use the extra information. This information is intended to argue against other processes and/or provide added support for the "correct" process. A plot of each of the four test cases, along with the represented process, is shown in Figure 7.

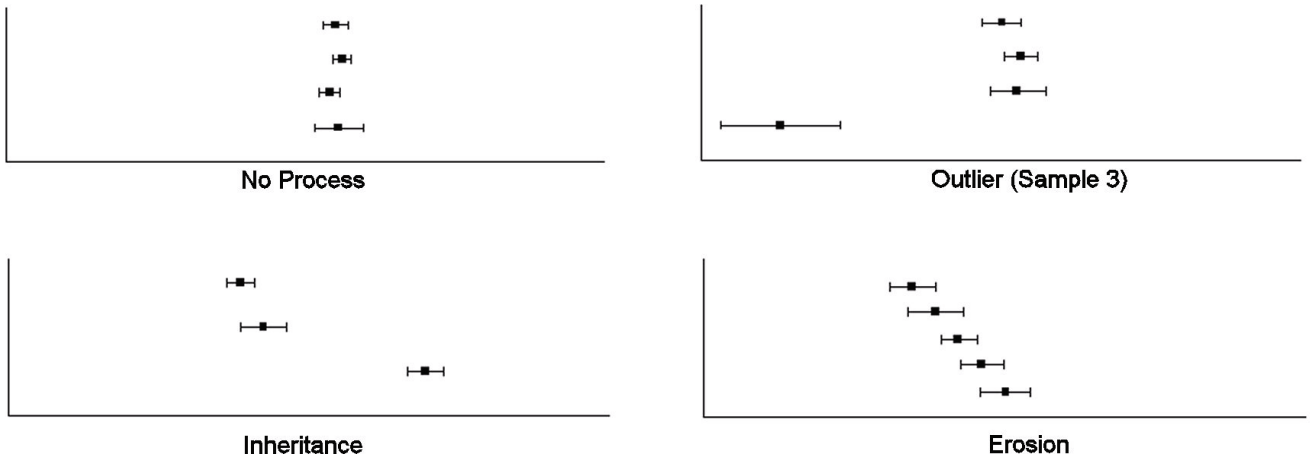

Figure 7: The four test cases used to validate the Calvin prototype. Each is labeled with the process represented by the samples. The system outputs the correct process for each of these cases.

Sections 8.1 and 8.2 contain the details of test cases two and four, the idealized examples of an outlier and erosion. For test case number two, Chris felt that the only reasonable explanation is an outlier. No other explanation accounts for the single sample much younger than the other three. This conclusion is supported by the fact that the outlier sample has a much larger error, noticeably different chemistry, was not embedded in the soil matrix of the moraine, and was physically far away from the other samples. Interestingly Chris's first statement about the process involved for this case was "two landforms," not only identifying one sample as an outlier but providing a theory that the outlier came from a different landform originally. For test case four, Chris was extremely confident that the correct process was erosion because of the linear sample distribution. The fact that the samples are from a moraine with a flat crest and fairly old provides good supporting evidence. When asked why inheritance was not a correct choice, he stated that there was no strong evidence for inheritance, although there is also no strong evidence against it. I then ran Calvin on the same data. The system's inputs and outputs for these two cases are presented below, along with a translation of the output and a discussion of how the system produces this reasoning.

8.1. Test Case 2 (Outlier)

Input:

Landform Type: Moraine

Sample	Age	Error	Chemistry	Location	Boulder Size	In Matrix?	Elevation
0	105000	5000	A	A	5	yes	3000
1	110000	4000	A	A	10	yes	3000
2	109000	7000	A	A	3	yes	3000
3	50000	15000	B	B	7	no	3000

Output:

Calvin's full output can be found in Appendix B. Calvin produces its output by printing simple representations of the rules contained in all the arguments it has produced. This output tends to closely resemble the source code, with some variable names replaced by values. This output is extremely difficult to read (and requires an examination of the code to fully understand); I will give a detailed text description of the output instead. I am currently working on a user interface that will contain output readable by ordinary users.

An outlier, the correct answer for this sample set, is a "low-priority" process (examined last by the engine), so Calvin produces arguments for all the processes currently implemented in its knowledge base. The engine always begins by examining "no process," erosion, and inheritance. "No process" is simple: this conclusion applies iff all samples are close to the mean sample age (where close is defined by the mean sample error). The next rule used takes the mean sample age and mean error and determines if all samples fall within the range determined by these two values. These rules are shown in Figure 8. Since the answer is no, this is a good argument against "no process." Calvin's confidence in this

```
(
  none
  ((argument
    ages-line-up))
  (scalar .95)
)

(
  ages-line-up
  ((calculation
    mean := calculate-mean age)
   (calculation
    std-dev := calculate-mean error)
   (argument
    all-within-range mean std-dev age))
  (selector 3 1)
)
```

Figure 8: Rules in the prototype version of Calvin expressing knowledge that "No process" applies iff all sample ages line up and that sample ages line up when all sample ages are within the mean error of the mean age.

argument is about -0.25; this is a case where the negative confidence in an argument should be much stronger.

Next Calvin considers erosion. Two rules in the database provide some evidence for erosion: the landform is relatively old and it is a moraine (see Figure 9). However, neither of these observations is especially *strong* evidence for erosion. On the other hand, several rules in the database allow the system to build stronger evidence *against* erosion. Specifically, the sample ages for this landform cannot be evenly spaced along a straight line and the moraine does not show visual signs of erosion (viz., it does not have a flat crest). Erosion is not strongly ruled out, however; this is a case where the ultimate version of Calvin might advise taking more samples to provide stronger evidence against a linear trend in ages.

The last process that Calvin must always consider is inheritance. Once again there is some support for this conclusion: inheritance is also somewhat more likely on a moraine, and the difference between the youngest and oldest samples does not exceed a theoretical maximum amount of inheritance (Lal 1991). On the other hand, if inheritance is affecting only a few samples, then it ought to be possible to remove these older samples and arrive at a "no process" conclusion on the modified dataset by simply building an argument for no process using the modified sample set. The system finds that this is not possible to do without removing most of the samples in the input set.

Since it has not yet found a convincing argument for some process, Calvin now considers the possibility that snow has caused the sample distribution. However, the system finds no evidence for snow and some evidence against it. In the case of snow falling on a moraine, it is expected that smaller boulders will have younger initial ages because their tops will be covered by snow for more of the year. The sample set does not follow this trend, so snow is determined to be an unlikely explanation.

Finally, Calvin considers each sample in turn as a possible outlier. The only evidence that the first sample might be an outlier is the fact that its error is significantly different from the average

```
(
  erosion
  ((calculation
    max-age := calculate-max age))
   (observation
    max-age > 50000)
  (selector 2 .3)
)
```

```
(
  erosion
  ((observation
    landform-type = moraine))
  (asymmetric-scalar (.2 . 0))
)
```

Figure 9: Rules providing evidence that the landform in Test Case 2 may have eroded because it is relatively old and because it is a moraine.

error of all the samples (this is because there are so few samples, and is not very compelling evidence). In contrast, this sample appears to have come from the same origin as most of the other samples in the set, because it is embedded in the soil matrix of the moraine and has the same general chemistry. Finally, removing only this sample does not allow the system to make a good argument for “no process,” so it is highly unlikely to be an outlier. The arguments considering the next two samples as possible outliers are extremely similar.

When Calvin considers the final sample as a possible outlier it finds quite a bit of evidence in support of this conclusion. Not only is the error for sample 3 significantly different from the other samples, it also has a different chemistry and is not embedded in the soil matrix of the moraine, making it quite likely to have come from a different source after the initial formation of the moraine. Finally, removing this sample from the set allows the system to argue successfully for “no process,” a powerful argument that this sample is a lone outlier. Thus the system concludes that the most convincing process is that sample 3 is an outlier, probably having come from a different landform after the initial formation of the moraine.

8.2. Test Case 4 (Erosion)

Input:

Landform Type: Moraine

Flat Crest

Sample	Age (yrs)	Error (yrs)	Chemistry	Elevation
0	95000	5000	A	4000
1	100000	6000	A	4000
2	105000	4000	A	4000
3	110000	4500	A	4000
4	115000	5500	A	4000

Output:

The full output for this test case can be found in Appendix C. As before, the Calvin engine considers the “no process” explanation first. Using the same procedure from test case 2, this conclusion is ruled out because the samples are distributed over a wider range than the average sample error. Erosion is considered next. In this test case there is significant evidence for erosion and no evidence against it. Rules in the system capture the fact that a linear pattern in the sample ages and a relatively old landform are good evidence for erosion. Also helpful is

the fact that this landform is a moraine. The system has also built a sub-argument that states that experts found visual signs of erosion while collecting samples, in this case that the crest of the moraine was flattened.

Although it has a strong argument for erosion, Calvin must still consider the evidence for and against inheritance. The fact that this landform is a moraine and that the inheritance amount is less than the theoretical maximum make inheritance a possibility. However, inheritance is unlikely since Calvin cannot remove only a small number of the oldest samples and then argue successfully for “no process.” Therefore erosion is by far the most reasonable conclusion for this sample set. Because the system can build an argument for erosion with a confidence higher than 0.5 (a value I have defined arbitrarily because it will not be needed once confidences are symbolic), it does not need to examine any further conclusions in this case.

One important weakness in both test cases is that the negative confidences are often too high (too small an absolute value). That is, even when experts would strongly reject some conclusion, Calvin may only weakly reject it. It is my belief that a well organized, symbolic system of confidence levels will ameliorate this problem.

9. Conclusion

Calvin is an argumentation system that will automate parts of the reasoning process of experts in cosmogenic isotope dating in an elegant and intuitive way. This is a challenging problem that has never before been addressed by a reasoning system of any kind. Although there is still much work to be done, the prototype version of the system is able to produce promising results.

The reasoning methods used by experts in isotope dating are informal and quixotic. They are generally heuristic-based and involve both contradictory data and contradictory knowledge. Furthermore, many of the heuristics used by experts seem to express tendencies and preferences rather than definite facts about the world. This type of reasoning is not especially difficult for human beings (though any given problem instance may be quite challenging), but it is extremely difficult for an automated system. Argumentation provides a flexible framework for accurately reproducing this type of reasoning. It is able to model uncertain heuristics and contradictory knowledge in a graceful way that closely mirrors the reasoning process that experts appear to use (and Calvin will be much more graceful once the confidence system is fixed).

There is still significant work to do to finish this thesis. As discussed in Sections 7 and 9, the system of confidence in use in the prototype system has significant weaknesses and should probably be completely replaced. This overhaul is probably the most difficult piece of work remaining, but I am making progress and have some ideas for a final confidence system. Also, more knowledge needs to be added to the system to completely capture experts' reasoning, and other improvements are needed before Calvin is finished.

The final version of the system will be useful to experts in isotope dating because it will reduce the effort they need to spend on redundant and boring work. Students of geology will be able to make use of the final version of the system to better understand how to reason about problems in isotope dating, especially by observing the reasoning process used by Calvin. The final system will demonstrate that the style of reasoning used by experts, difficult to model in a traditional AI framework, can be cleanly and clearly encapsulated in an argumentation system in a form that closely mirrors experts' internal knowledge representations.

10. References

Anderson, K., Bradley, L., Zreda, M., Rassbach, L., Zweck, C., and Sheehan, E. 2007. ACE: Age Calculation Engine—A Design Environment for Cosmogenic Dating Techniques (Forthcoming). In Proceedings of the First International Conference on Advanced Engineering Computing and Applications in Sciences. Tahiti.

Bouwer, A. and Bredeweg, B. 2002. Aggregation of Qualitative Simulations for Explanation. In Proceedings of the Sixteenth International Workshop on Qualitative Reasoning. Barcelona.

Boy G. and Gruber, T.R. 1990. Intelligent Assistant Systems: Support for Integrated Human-Machine Systems. In Proceedings of 1990 AAAI Spring Symposium on Knowledge Based Human-Computer Communication. Stanford University.

Bradwell, T. 2001. A New Lichenometric Dating Curve For Southeast Iceland. Geografiska Annaler: Series A, Physical Geography 83(3): 91-101.

Brashers, D., Adkins, M., Meyers, R., Mittleman, D. 1995. The Facilitation of Argumentation in Computer Mediated Group Decision-Making Interactions. In Proceedings of the Third ISSA Conference on Argumentation.

Cem Say, A.C. 1999. Using Inter-Behavior Contradictions for Modeling and Diagnosis. In Proceedings of the Thirteenth International Workshop on Qualitative Reasoning. Loch Awe, Scotland.

Clark, P.E. 1989. Exemplar-Based Reasoning in Geological Prospect Appraisal, Technical Report, 89-034, Turing Institute, University of Stratchclyde.

CRONUS 2007. Information about the CRONUS Earth project is available at: http:// www.physics.purdue.edu/cronus/index.shtml

Cunningham, P., Bonzano, A., and Smyth, B. 1995. An Incremental Case Retrieval Mechanism for Diagnosis. Technical Report TCD-CS-95-01, Department of Computer Science, Trinity College.

Desilets, D. and Zreda, M. 2006. Elevation Dependence of Cosmogenic 36Cl Production in Hawaiian Lava Flows. Earth and Planetary Science Letters 246: 277-287.

Doyle, J. 1983. Methodological Simplicity in Expert System Construction: The Case of Judgments and Reasoned Assumptions. AI Magazine 4(2): 39-43.

Dung, P.M. 1995. On the Acceptability of Arguments and its Fundamental Role in Nonmonotonic Reasoning, Logic Programming, and N-Person Games. Artificial Intelligence 77(2): 321-357.

Etherington, D., Kraus, S., and Perlis, D. 1991. Nonmonotonicity and the Scope of Reasoning. Artificial Intelligence 52(3): 221-261.

Farley, A.M. 1997. Qualitative Argumentation. In Proceedings of the Eleventh International Workshop on Qualitative Reasoning. Cortona, Italy.

Forbus, K.D. and De Kleer, J. 1993. *Building Problem Solvers*. MIT Press.

Gaines, B.R. 1989. An Ounce of Knowledge is Worth a Ton of Data: Quantitative Studies of the Trade-off Between Expertise and Data Based on Statistically Well-Founded Empirical Induction. In Proceedings of the Sixth International Workshop on Machine Learning,156-159.

Gaines, B.R. 1996. Transforming Rules and Trees into Comprehensible Knowledge Structures. In Advances in Knowledge Discovery and Data Mining, 205-226. Cambridge, MA: MIT Press.

Gosse, J. C. and Phillips, F. M. 2001. Terrestrial Cosmogenic Nuclides: Theory and Application. Quaternary Science Reviews 20(14): 1475-1560.

Kolodner, J. 1993. *Case-Based Reasoning*. Morgan Kaufmann.

Krause, P., Ambler, S., Elvang-Gøransson, M., and Fox, J. 1995. A Logic of Argumentation for Reasoning Under Uncertainty. Computational Intelligence 11: 113-131.

Lal, D. 1958. Investigation of Nuclear Interactions Produced by Cosmic Rays. Ph.D. diss., University of Bombay.

Lal, D. 1991. Cosmic Ray Labeling of Erosion Surfaces: *In Situ* Nuclide Production Rates and Erosion Models. Earth and Planetary Science Letters 104: 424-439.

Lucas, P.J.F. 1997. Symbolic Diagnosis and its Formalisation. The Knowledge Engineering Review 12(2): 109-146.

McCarthy, J. 1980. Circumscription—a Form of Non-Monotonic Reasoning. Artificial Intelligence 13(1,2): 27-39, 171-172.

McCarthy, J. 1986. Applications of Circumscription to Formalizing Common Sense Knowledge. Artificial Intelligence 26(3): 89-116.

McIlraith, S. and Reiter, R. 1992. On Tests for Hypothetical Reasoning. In Readings in Model-Based Diagnosis, 89-96. San Mateo, CA: Morgan Kaufman.

Parsons, S., Sierra, C., and Jennings, N. 1998. Multi-context Argumentative Agents. In Proceedings of the Fourth International Symposium on Logical Formalizations of Commonsense Reasoning. London.

Pearl, J. 1988. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann.

Pearl J. 1999. Reasoning with Cause and Effect. In Proceedings of the Sixteenth International Joint Conference in Artificial Intelligence, 1437-1449. Stockholm, Sweden.

Pereira, L.M., Alferes, J.J., and Apar'icio, J.N. 1991. Nonmonotonic Reasoning with Well Founded Semantics. In Proceedings of the Eighth International Logic Programming Conference, 475-489. Paris: MIT Press.

Pollock, J.L., 1994. Justification and Defeat. Artificial Intelligence 67(2): 377-407.

Prakken, H. 1996. Dialectical Proof Theory for Defeasible Argumentation with Defeasible Priorities. In Proceedings of the Biannual International Conference on Formal and Applied Practical Reasoning Workshop, 202-215. Bonn.

Puyol-Gruart, J., Godo, L., and Sierra, C. 1992. A Specialization Calculus to Improve Expert Systems Communication. In Proceedings of the European Conference on Artificial Intelligence, 144-148.

Reiter, R. 1980. A Logic for Default Reasoning. Artificial Intelligence 13: 81-132.

Santos, E. 1991. On the Generation of Alternative Explanations with Implications for Belief Revision. In Proceedings of the Seventh Conference on Uncertainty in Artificial Intelligence, 339-347. San Mateo, CA: Morgan Kaufmann.

Shanahan, T.M. and Zreda, M. 2000. Chronology of Quaternary Glaciations on Mount Kenya and Kilimanjaro. Earth and Planetary Science Letters 177: 23-42.

Shaw, M.L.G. and Gaines, B.R. 1986. Interactive Elicitation of Knowledge from Experts. Future Computing Systems, 1(2): 151-190.

Stolle, R. and Bradley, E. 1996. A Customized Logic Paradigm for Reasoning about Models. In Proceedings of the Tenth International Workshop on Qualitative Reasoning. Stanford Sierra Camp, CA.

Struss, P. 2004. Deviation Models Revisited. In Proceedings of the Eighteenth International Workshop on Qualitative Reasoning. Evanston, IL.

Surma, J., and Vanhoof, K. 1995. Integrating Rules and Cases for the Classification Task. In Proceedings of the First International Conference on Case-Based Reasoning, 325-334. Berlin: Springer Verlag.

Turner, R.M. 1992. A View of Diagnostic Reasoning as a Memory-Directed Task. In Proceedings of the Fourteenth Annual Conference of the Cognitive Science Society. Bloomington, IN: Cognitive Science Society.

Vreeswijk, G. 1991. The Feasibility of Defeat in Defeasible Reasoning. In Proceedings of the Second International Conference on Principles of Knowledge Representation and Reasoning, 478-483. Morgan Kaufmann.

Vreeswijk, G. 1993. Defeasible Dialectics: a Controversy-Oriented Approach Towards Defeasible Argumentation. Journal of Logic and Computation 3: 317-334.

Zreda, M.G., Phillips, F.M., and Elmore, D. 1994. Cosmogenic 36Cl Accumulation in Unstable Landforms. Water Resources Research 30: 3127-3136.