
Using Equations in Concept Maps to Graphically Build Knowledge Bases

Aaron Spaulding

SRI International
333 Ravenswood Avenue
Menlo Park, CA 94025
spaulding@ai.sri.com

Vinay K. Chaudhri

SRI International
333 Ravenswood Avenue
Menlo Park, CA 94025
chaudhri@ai.sri.com

Bonnie E. John

Human-Computer Interaction
Institute
Carnegie Mellon University
Pittsburgh PA 15213
bej@cs.cmu.edu

Gus Prevas

Independent Consultant
Menlo Park, CA
kprevas@gmail.com

Sunil Mishra

SRI International
333 Ravenswood Avenue
Menlo Park, CA 94025
smishra@ai.sri.com

John Pacheco

SRI International
333 Ravenswood Avenue
Menlo Park, CA 94025
pacheco@ai.sri.com

Abstract

Given an equation representing motion in one dimension, $v_2 = v_1 + a * t$, and the value of any of the three variables, it is straightforward to compute the fourth variable by using a pocket calculator or a simple program. But a simple program or a calculator does not understand anything about velocity, acceleration or time, nor how to determine their values in a given scenario. Such lack of knowledge prevents the use of an equation for any sort of reasoning. In this paper we describe a user interface to connect equations to richly defined concepts within a knowledge base. This makes it possible to support reasoning about the variables referenced in an equation. An evaluation of our system found that users could successfully build knowledge bases capable of using inference to answer questions. We discuss the lessons learned during the process.

Keywords:

Interaction design, contextual inquiry, equations, concept maps, knowledge representation and reasoning, AURA

ACM Classification Keywords

Graphical User Interfaces (H5.2), Interaction Techniques (I.3.6), User-centered Design (H5.2)

Introduction

Traditional information retrieval systems are not well-suited for scientific knowledge management because they cannot reason with the subject matter. Knowledge management systems generally require highly trained knowledge engineers to build effective knowledge bases that can support reasoning. In an effort to enable users to author knowledge bases with minimal training in knowledge representation, we have been constructing a knowledge acquisition tool called Automated User-Centered Reasoning and Acquisition System (AURA). This work is part of a long-term research program called Project Halo funded by Vulcan Inc [1]. The goal of Project Halo is to construct an application called Digital Aristotle that is capable of answering questions and explaining them on a wide variety of science topics.

As a near-term goal, we are focusing on acquiring knowledge from 50 pages of science text books in the domains of Physics, Chemistry, and Biology such that

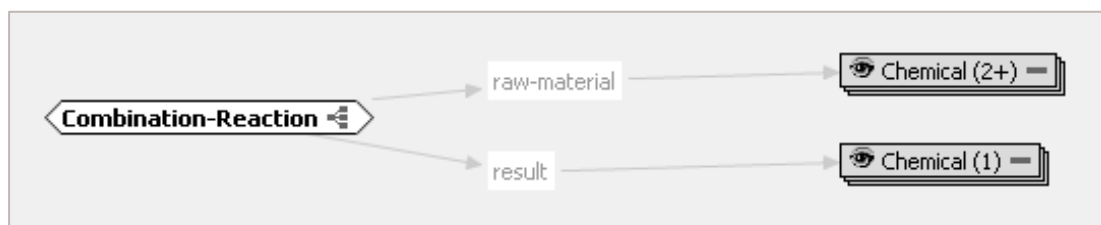


Figure 1: AURA representation of a Combination Reaction. We can interpret this figure as follows: For every instance of a *Combination Reaction*, there exists a two or more *Chemicals* -- visually indicated by a label of (2+) -- that is its *raw material*, and a *result* that is constrained to exactly one *Chemical*. The stacked node used for both *Chemicals* further emphasizes that a constraint has been applied to the node and relation. In AURA, the relation names such as *result*, and *raw-material* are a part of the Component Library knowledge base. The nodes such as *Chemical* are a part of the pre-existing domain-specific knowledge base that is expandable by the user.

the system can answer questions similar to those in an Advanced Placement exam. Moreover, we wish to ensure that the physicists, chemists, and biologists can enter the knowledge themselves, without requiring extensive training or the assistance of knowledge engineers.

The AURA System

AURA enables the acquisition of knowledge that can be used for declarative inference. For example:

- A Combination Reaction is a kind of Chemical Reaction
- In a Combination Reaction, two or more Chemicals combine to form exactly one Chemical
- The pH of a solution can be computed as $\text{pH} = \log [\text{H}^+]$, where $[\text{H}^+]$ represents the concentration of Hydrogen ions in a solution.

Using this knowledge, AURA can answer questions about different kinds of reactions, their properties, and whether a given reaction is of a specific type, and perform computation for the pH of a solution.

AURA uses the KM system for representing and reasoning with the acquired knowledge [2]. The knowledge acquisition in AURA is driven by a knowledge base called the Component Library [3]. The Component Library is built by knowledge engineers and contains domain-independent classes such as *Attach*, *Penetrate*, *Physical Object*; predefined sets of relations such as *agent*, *object*, *location*; and property values to help represent units and scales such as *size*, *color*, *distance*, etc. We present a class to a user as a graph called a concept map (C-Map) that represents an

```

(_System32 has
  (instance-of (System))
  (component (_Move34))
  (equation (_Equation-Set33)))

(_Equation-Set33 has
  (instance-of (Equation-Set))
  (equation-symbol
    ( (:pair t _Duration
         Value41)
      (:pair a _Acceleration-Value36)
      (:pair v_1 _Velocity-Value39)
      (:pair v_2 _Velocity-Value40)))
  (equation-expression
    ((QUOTE
      (= v_2 (+ v_1 (* a t)))
    )))
  (equation-uses
    (_Duration-Value41
     _Acceleration-Value36
     _Velocity-Value39
     _Velocity-Value40)))

```

Figure 2: code for adding the equation $v_2 = v_1 + a * t$ to a concept describing Move. With this equation represented in the KB, AURA can infer facts such as (1) if the motion involves a free fall v_1 must be equal to 0 and (2) if the motion is under gravity, the acceleration value must be $9.8m/s^2$.

example of the class (Figure 1), and the users construct new classes by connecting instances of the graphs representing existing classes using graph operations such as *add*, *connect*, *specialize*, etc. [4].

AURA can capture knowledge about concept taxonomies, deductive rules, constraints, chemical reactions, equations, and tables. Users can pose questions to the resultant knowledge base using simplified English for initial formulation followed by a graphical editing of the system's interpretation of the question.

Prior to the user interface we describe here, knowledge about equations could only be added as code in the KM language (Figure 2). To find a better solution for integrating equations and C-Maps, we turned our focus to the potential users. What tools do they use? How do they work with equations? What formal representations of knowledge are familiar to them?

Target Users

AURA is designed for two types of users. One possible model for constructing large databases of scientific knowledge is to support science graduate students on "knowledge authoring assistantships" akin to research or teaching assistantships. Therefore, for authoring knowledge AURA's target users are graduate students in Biology, Chemistry and Physics. One group of potential consumers of the knowledge are undergraduate students learning science, therefore, AURA's question asking component is targeted for undergraduate students in the same domain.

Contextual Inquiry

To understand our target users, we conducted contextual inquiries (CIs) with three graduate students and two post-docs in Biology, Chemistry and Physics, and created models and personae from the data. The CIs showed that the users had no preference for a particular tool for creating equations (save by hand). While every user had their own computer, the only time they used equation editor software was when preparing a technical paper, as CI2 put it "to make things look pretty." Generally LaTeX or MathType was used, depending on the submission requirements of the paper. One user had some familiarity with Mathematica, but this was for a specific course he had taken, and he had not used it in the time since. The users were sensitive to the extra effort required to use equation editors, saw no advantage, and did not use them unless required. There have been a limited number of prior systems that combine equations with reasoning [5], but they have not been widely deployed, and were unfamiliar to our users.

Competitive Analysis

We performed a competitive analysis of the equation editors mentioned during the contextual inquiries and their competitors. Three of the systems (LaTeX [6], MathType [7], and Equation Illustrator V [8]) were equation typesetting tools, and did not include any mathematical parsing of the equations. The systems that could parse equations (Scientific Notebook [9] and Graphing Calculator [10]) provided a much more complex feature set than needed by AURA. Two other applications (Mathematica [11] and Maple [12]) had excellent expressivity and ability to perform complex mathematical operations, but were rich programming environments, and too complex to integrate with AURA.

$$\begin{aligned} \Delta^2 &= \left| \left(\gamma \partial_u - \frac{g}{2} \tau \cdot W_u - \frac{g'}{2} B_u \right) \phi \right|^2 - V(\phi), \quad \phi = \begin{pmatrix} 0 \\ \phi(u) \end{pmatrix} \\ &= \left| \begin{pmatrix} 0 \\ \gamma \partial_u \phi \end{pmatrix} - \frac{g}{2} \begin{pmatrix} W_u^1 - \gamma W_u^2 \\ -W_u^3 \end{pmatrix} \phi - \frac{g'}{2} \begin{pmatrix} 0 \\ B_u \end{pmatrix} \phi \right|^2 \\ &= \left| \begin{bmatrix} -\frac{g}{2} (W_u^1 - \gamma W_u^2) \\ \gamma \partial_u + \frac{g}{2} W_u^3 - \frac{g'}{2} B_u \end{bmatrix} \phi \right|^2 \\ &= \left[-\frac{g}{2} (W_u^1 + \gamma W_u^2), -\gamma \partial_u + \frac{g}{2} W_u^3 - \frac{g'}{2} B_u \right] \phi^\dagger \cdot \begin{bmatrix} -\frac{g}{2} (W_u^1 - \gamma W_u^2) \\ \gamma \partial_u + \frac{g}{2} W_u^3 - \frac{g'}{2} B_u \end{bmatrix} \phi \end{aligned}$$

1/29 H₂O₂ standardized

- 30% H₂O₂ diluted 100 fold
- A @ 230nm measured with blank of 2 mL H₂O
- $\epsilon_{H_2O_2} @ 230nm = 72.8 \text{ cm}^{-1} \text{ M}^{-1}$

$A = \epsilon l c$
 $l = \text{path length} = 1 \text{ cm}$
 $\epsilon = 72.8 \text{ cm}^{-1} \text{ M}^{-1}$
 $A = \text{from experiment}$

Figure 3: Top: excerpt of CI 2 current work. A physicist, much of his time was spent working out calculations by hand.

Bottom: excerpt from a page of a lab book from CI 5 showing equations integrated with other information.

While the analysis of existing tools showed that none could be easily integrated into AURA, they had features we felt were useful: WYSIWYG entry of equations, and automatic formatting.

The Equation Editor Design Principles

We incorporated well known HCI principles in our design approach for AURA. Based on the contextual inquiries and competitive analysis, we focused on the following principles:

Naturalness in Representation

The contextual inquiries showed that the traditional representation of equations is the most natural for the users. It is what they use, and what is printed in textbooks. Most equation editors we reviewed supported this natural form either throughout the process or as an output format. In AURA, we attempt to meet the user's expectations by automatically formatting the equations, and allowing them to enter any variables that they prefer, e.g., v_1 , v_i , or $v_{initial}$ for initial velocity. Permitting this naturalness has the benefit that equations correctly entered from a source text look exactly like the version in the text. Any entry errors would stand out immediately, and can be quickly corrected by the user.

Efficiency

We saw that users were not enamored by existing equation editors, and were used to having the flexibility that a pen and paper provide. They were used to writing equations quickly – they cannot be stuck clicking on a virtual calculator keyboard, or spending time fussing with formatting. The analysis of existing tools uncovered common key sequences across some of the applications, such as a caret (^) to indicate an

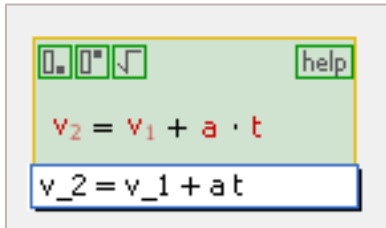


Figure 4: Final version of the equation editor showing user-entered equation and formatted version. Note the equation editor automatically adds the multiplication operator. The variables are red to indicate that they are not yet mapped to concepts in the knowledge base.

exponent. We incorporated these standards to leverage any previous experience that the users may have had with equation editors. Automatic formatting of expressions saves the users from undue time wasted to “make them look pretty.”

Simple and Direct Connections to Concept Maps

The C-Maps are designed to make the state of the knowledge base clearly visible, and to allow users to alter it through direct manipulation of the graph. Several early rounds of user tests showed that the users found the C-Maps easy to understand, create, and manipulate. In developing the equation editor we wanted to integrate the expressiveness of equations into the C-Map structure, while adding a minimal amount of complexity. We did this by expanding the link and node metaphor to attach meaning to the equation variables. The users enter an equation, using any variables that they prefer. The system recognizes the variables and highlights them until they have been mapped to a concept (see figure 4). To complete this mapping, the user simply drags a link from the appropriate concept to the variable. This only needs to be done once per variable; other occurrences of that variable within the same C-Map will be automatically understood by the system. Once a concept node is assigned to a variable, the node is labeled with the variable, and connected to the equation with a line (figure 5). These lines highlight with mouse-over of the concept or equation, reinforcing the connection. To help avoid visual clutter, the connecting lines can be hidden.

User Testing and Design Iterations

Early user tests showed that the auto-formatting and error checking caused major problems. The initial implementation rigorously parsed the user input to

ensure that entered equations were mathematically sound. However, users were continuously frustrated by the lack of flexibility permitted by this parser. Users wanted to include partial equations for later expansion, and to incomplete or conflicting versions of the equations as a note to themselves. In short, they wanted the flexibility of paper.

Final Design

We loosened the error checking to accommodate unexpected entry patterns. We added an unformatted version of the equation to create a linear version of the equation to resolve difficulties in editing equations with complex representations such as compound fractions. Tests on the final version showed that revising equations was much easier, and that users were not confused by seeing both the linear and formatted version of the equation.

Training and Evaluation

We analyzed the equation editor in the context of an extensive evaluation of the effectiveness of AURA as a whole in June of 2006. Six users received 40 hours of training each, and then used AURA for 80 hours to encode knowledge about physics, chemistry, and biology. The specific training for the equation editor was limited to roughly 30 minutes. A separate set of six users then spent one week posing questions to AURA.

During the use of the system, we tracked any help requests made by the users, and none of the requests were related to the use of the equation editor. This attests to the usability success of the equation editor. As expected, the equation editor was most relevant in the physics domain, where nearly all questions asked required AURA to reason with equations in order to

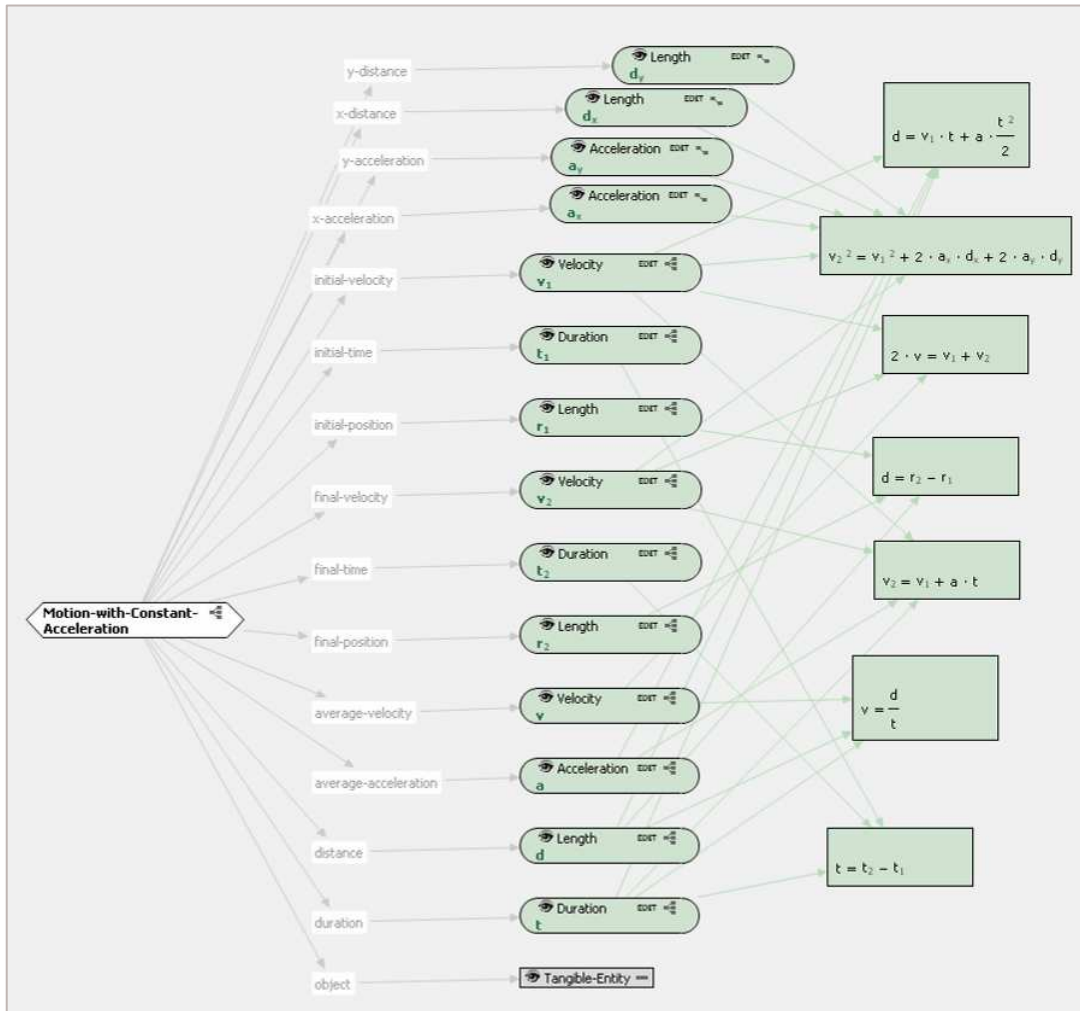


Figure 5: The C-Map representation of Motion with Constant Acceleration authored by a user. Note that the equations are all fully linked to the C-Map. When the user mouses over one of the equations, lines between the equation and related nodes highlight to visually strengthen the connections. Also, each property node (Acceleration, Velocity, etc.) has a label to connect it to the corresponding variable used in the equations.

provide an answer. The physics users added approximately 25 equations to the system with an average equation creation time of about 2 minutes. The average time to completely create a concept (e.g., Motion with Constant Acceleration) was around 20 minutes. In that context, the time taken for using the equation editor is very reasonable.

The knowledge bases created by the users were able to correctly answer 49 questions. Some examples of questions correctly answered are:

- A light plane must reach a speed of 30 m/s for takeoff. How long a runway is needed if the (constant) acceleration is 3.0 m/s²?
- A stone is dropped from the top of a cliff. It is seen to hit the ground below after 3.50 s. How high is the cliff?

In each case, AURA had to map the English word to a relevant concept in the knowledge base – e.g., a stone is a physical object; to drop something means it falls. Then AURA needs to determine the appropriate model to use – for instance, Motion with Constant Acceleration (Figure 5) could be used with the first question above, but a different model (such as Free-Fall) that specifies acceleration due to gravity would need to be used to answer the second question. AURA then picks an equation within the model to use in solving the question and generating an answer.

Conclusions and Future Work

We can draw the following general lessons from this experience:

Leverage known interactions: We were able to leverage learned C-Map interactions and common practices found in the comparison of equation formatting tools to create a simple interface that was easy for the users to use – and they were able to create sophisticated and effective models.

Allow incomplete knowledge and support revision: We needed to allow users to enter partial equations that could be completed later. This turned out to be a crucial feature.

Multiple forms of feedback: Providing multiple forms of user feedback – one in which we showed the input keystrokes, and the other in which we showed a formatted representation – was effective.

Understand users, test, and iterate: The contextual inquiries we conducted gave us a basis for our early designs, but user testing and design iteration were essential to developing a useable system. Our initial focus on always presenting equations in a natural form was based on data from the CI's but did not directly translate to an implementation in AURA. User tests showed that the hybrid system with simultaneous natural and input keystrokes was much easier for users to edit and revise.

The user evaluation and analysis of the user models have exposed areas for future work in the areas of simplifying expressions and expanding the system expressiveness.

Simplifying Multiple Dimensions in an Equation

A physics equation such as $v = \frac{1}{2} \cdot at$ can be applicable to multiple dimensions. For example, we can write

$v_y = \frac{1}{2} \cdot a_y t$, $v_x = \frac{1}{2} \cdot a_x t$ for y and x dimensions, respectively. We must find a way to make it visible to the user that the equation can be applicable to multiple dimensions even though it may be written without any specific mention of dimension.

Expansion of Equation Editor Expressiveness

Many equations in physics require using vectors and vectors are not supported in the current system. For example, consider the equation: $\vec{v}_x + \vec{v}_y = \vec{v}$. We cannot author this equation using the current equation editor, and will need to extend the interface to represent vectors.

Acknowledgments

We thank the members of the AURA development team at the University of Texas at Austin and The Boeing Company for their contributions to the development of AURA. This work has been funded by Vulcan Inc.

References

- [1] Friedland, N.S., et al., Project Halo: Towards a Digital Aristotle. AI Magazine, 2004. 25(4): p. 29-48.
- [2] Clark, P. and B. Porter. KM - The Knowledge Machine: Users Manual. 1999.
- [3] Barker, K., B. Porter, and P. Clark, A Library of Generic Concepts for Composing Knowledge Bases, in Proc. 1st Int Conf on Knowledge Capture (K-Cap'01). 2001. p. 14-21.
- [4] Clark, P., et al., Knowledge Entry as the Graphical Assembly of Components, in Proc. 1st Int Conf on Knowledge Capture (K-Cap'01). 2001. p. 22-29.
- [5] Forbus, K., et al., CyclePad: An articulate virtual laboratory for engineering thermodynamics. Artificial Intelligence, 1999. 114: p. 297-347.

- [6] LaTeX.
<http://www.latex-project.org/>.
- [7] MathType 5.
<http://www.dessci.com/en/products/mathtype/>.
- [8] Equation Illustrator V.
<http://www.equation-illustrator.com/>
- [9] Scientific Notebook 5.5.
<http://www.mackichan.com/products/snb.html>.
- [10] Graphing Calculator
<http://www.pacifict.com/Products.html>
- [11] Mathematica 5.2
<http://www.wolfram.com/products/mathematica/>.
- [12] Maple 10.
<http://www.maplesoft.com/>.