Repairing Incorrect Knowledge with Model Formulation and Metareasoning

Scott E. Friedman & Kenneth D. Forbus
Qualitative Reasoning Group, Northwestern University
2145 Sheridan Road, Evanston, IL 60208, USA
{friedman, forbus}@northwestern.edu

Abstract

Learning concepts via instruction and expository texts is an important problem for modeling human learning and for making autonomous AI systems. This paper describes a computational model of the self-explanation effect, whereby conceptual knowledge is repaired by integrating and explaining new material. Our model represents conceptual knowledge with compositional model fragments, which are used to explain new material via model formulation. Preferences are computed over explanations and conceptual knowledge, along several dimensions. These preferences guide knowledge integration and question-answering. Our simulation learns about the human circulatory system, using facts from a circulatory system passage used in a previous cognitive psychology experiment. We analyze the simulation's performance, showing that individual differences in sequences of models learned by students can be explained by different parameter settings in our model.

1 Introduction

Learning scientific concepts from instruction and expository texts is a familiar task for humans, but an unsolved problem in Artificial Intelligence. Cognitive science research has shown that when people learn from expository texts, they can repair flawed domain knowledge by self-explaining new material to themselves [Chi, 2000]. This *self-explanation effect* is a key component of learning via reading and instruction, since novices often possess misconceptions prior to learning. This prompts us to ask two questions:

1. How can new material be integrated into an existing, possibly flawed, domain theory?
2. Can we model the self-explanation effect to repair incorrect domain knowledge in an automated learning system?

This paper presents a model of the self-explanation effect. Our system uses qualitative *model fragments* [Falkenhainer and Forbus, 1991] to represent domain knowledge. When the system encounters new instructional material, it integrates it by (1) formulating qualitative models to explain the new material using model fragments and propositions, (2) finding contradictions between explanations, and (3) using preferences between explanations and model fragments to resolve contradictions and to guide knowledge integration. The system's knowledge is organized using the knowledge-based network of Friedman and Forbus [2010].

We simulate results from the cognitive science literature concerning the self-explanation effect in learning about the human circulatory system. In each simulation trial, the system begins with one of six mental models of the circulatory system found in students from Chi *et al* [1994]. We incrementally provide the system with facts from the circulatory passage used in the study, encoded as relational facts using an extended OpenCyc[1] ontology. The system performs self-explanation to integrate the new facts and incrementally revises its preferred model of the circulatory system. We assess learning with a subset of the pretest and posttest from Chi et al [1994] where the system plots the flow of blood in the body in a directed graph, identifies the concentration of chemicals (e.g. Oxygen, CO_2) in the blood at all points, and plots influences between quantities. We compare the system's performance to students, showing that most individual differences can be captured by different parameter settings of our model.

We begin by summarizing related work. We then discuss our knowledge representation, explanation-based knowledge organization, and model formulation. We present simulation results, and discuss future work.

1.1 Explanation-Based Learning & Belief Revision

We build upon a long history of learning by constructing explanations within Artificial Intelligence. Many systems that perform Explanation-Based Learning (EBL) [DeJong, 1993] create new knowledge by *chunking* explanation structure into a single rule [Laird *et al.*, 1987]. This speeds up future reasoning, but chunking alone does not change the deductive closure of the knowledge base. The self-explanation effect, by contrast, can lead to changes in the student's models, which chunking alone cannot capture.

Winston and Rao [1990] describe methods for using explanations to repair error-prone knowledge in classifying

[1] http://www.opencyc.org

Friedman, S. E. and Forbus, K. D. (2011). Repairing Incorrect Knowledge with Model Formulation and Metareasoning. Proceedings of the 22nd International Joint Conference on Artificial Intelligence. Barcelona, Spain.

artifacts, where explanations are trees of *if-then* rules over artifact features. Upon misclassification, the system analyzes its explanations and creates censor rules to prevent future misclassification. Like their system, our model diagnoses inconsistencies within and across explanations in its analysis, but it encodes epistemic preferences (rather than censors) to resolve these issues.

The CASCADE system [VanLehn *et al.*, 1992] has modeled the self-explanation effect on learning procedural problem-solving rules and control knowledge. However, it does not model the repair of conceptual knowledge.

Previous research in AI has produced postulates for revising beliefs within a knowledge base. The AGM postulates [Alchourròn *et al*, 1985] describe properties of rational revision operations for expansion, revision, and contraction of propositional beliefs within a deductively-closed knowledge base. Katsuno and Mendelzon's [1991] theorem equates these postulates to a revision mechanism based on total preorders over prospective KB interpretations. As shown below, our system computes a partial order over explanations rather than over propositional belief sets. Consequently, the granularity of consistency of our approach differs from these accounts of belief revision: it does not ensure a consistent, deductively-closed KB, but it does ensure consistency within explanations.

2. Representing & Building Qualitative Models

Reasoning about dynamic systems, such as the human circulatory system, makes several demands on our choice of knowledge representation. Consider, for example, representing the flow of blood from the heart to the body with the flawed "single loop" circulatory system model (Figure 4) frequently exhibited by novices. Even this oversimplified model requires representing mass entities (e.g. blood), generic concepts (e.g. contained fluid), entities (e.g. heart), processes (e.g. fluid flow), quantities (e.g. the blood concentration of CO_2 within the body), and influences between quantities (e.g. the CO_2 concentration of blood leaving the heart is qualitatively proportional to that of the blood within the heart). We review model fragments and model formulation, which are our methods of representing and assembling conceptual knowledge, respectively.

2.1 Model Fragments & QP Theory

Model fragments [Falkenhainer and Forbus, 1991] represent entities and processes, such as the fluid in a container, and the flow of fluid out of that container, respectively. For example, modeling heart-to-body blood flow within the "single loop" model involves several model fragments. Figure 1 shows two model fragment types used in the simulation: the entity `ContainedFluid`, and the process `FluidFlow`. Both have several components: (1) *participants* are the entities involved in the phenomenon; (2) *constraints* are statements that must hold over the participants in order to *instantiate* the model fragment as a distinct entity; (3) *conditions* are statements that must hold for the instance to be *active*; and (4) *consequences* are statements that hold when the instance is active.

```
PhysicalModelFragmentType ContainedFluid
Participants:
 ?con Container (containerOf)
 ?sub StuffType (substanceOf)
Constraints:
 (physicallyContains ?con ?sub)
Conditions:
 (greaterThan (Amount ?sub ?con) Zero)
Consequences:
 (qprop- (Pressure ?self) (Volume ?con))

QPProcessType FluidFlow
Participants:
 ?source-con Container (outOf-Container)
 ?sink-con Container (into-Container)
 ?source ContainedFluid (fromLocation)
 ?sink ContainedFluid (toLocation)
 ?path Path-Generic (along-Path)
 ?sub StuffType (substanceOf)
Constraints:
 (substanceOf ?source ?sub)
 (substanceOf ?sink ?sub)
 (containerOf ?source ?source-con)
 (containerOf ?sink ?sink-con)
 (permitsFlow ?path ?sub
              ?source-con ?sink-con)
Conditions:
 (unobstructedPath ?path)
 (greaterThan (Pressure ?source)
              (Pressure ?sink)))
Consequences:
 (greaterThan (Rate ?self) Zero)
 (i- (Volume ?source) (Rate ?self))
 (i+ (Volume ?sink) (Rate ?self))
```

Figure 1: *ContainedFluid* (above) and *FluidFlow* (below) model fragment types.

Several statements in Figure 1 use relationships between quantities from qualitative process (QP) theory [Forbus, 1984]. The relations `i+` and `i-` assert *direct influences* that describe derivative constraints on quantities, e.g. between a rate quantity `(Rate ?self)` of `FluidFlow` and an affected quantity `(Volume ?source)`. In this example, `(Volume ?source)` and `(Volume ?sink)` will be decreasing and increasing by `(Rate ?self)` of a `FluidFlow`. Further, the relations `qprop` and `qprop-` assert monotonic *indirect influences*. In Figure 1, the `qprop-` relation asserts that all else being equal, decreasing `(Volume ?con)` will result in `(Pressure ?self)` of a `ContainedFluid` increasing.

2.2 Model Formulation

Given a domain theory described by model fragments and a relational description of a scenario, the process of *model formulation* automatically creates a model for reasoning about the scenario [Falkenhainer & Forbus, 1991]. Our approach uses a back-chaining algorithm (similar to [Rickel and Porter, 1997]) to build scenario models. The algorithm is given (1) a domain theory (DT) that contains relations over entities (e.g. `(physicallyContains heart Blood)`, `(isa heart Heart)`), and (2) a target phenomenon to explain (e.g. blood flowing from the heart to the body, described as fact nodes f_{23-26} in Figure 3). The model formulation algorithm (**DoModelFormulation** in Figure 2) begins by finding model fragment types that are specializations of the type of target phenomenon (e.g. FluidFlow is a specialization of PhysicalTransfer). Next, it binds the variable participants of the model fragment using assertions from the scenario (e.g. ?sub → Blood, ?source-con → heart, and ?sink-con → body). If other participants are still unbound, the algorithm works backwards recursively, instantiating the participants using model fragments, constrained by the parent model fragment (e.g. a Contained-Fluid bound to ?source is recursively instantiated using participant bindings ?con → heart and ?sub → Blood). In the event that a participant cannot be found in the scenario, a skolem term may be created to assume its existence. This signifies a gap in the domain theory. This algorithm produces justification structure for the target phenomenon, in the form of model fragment instantiations and activations.

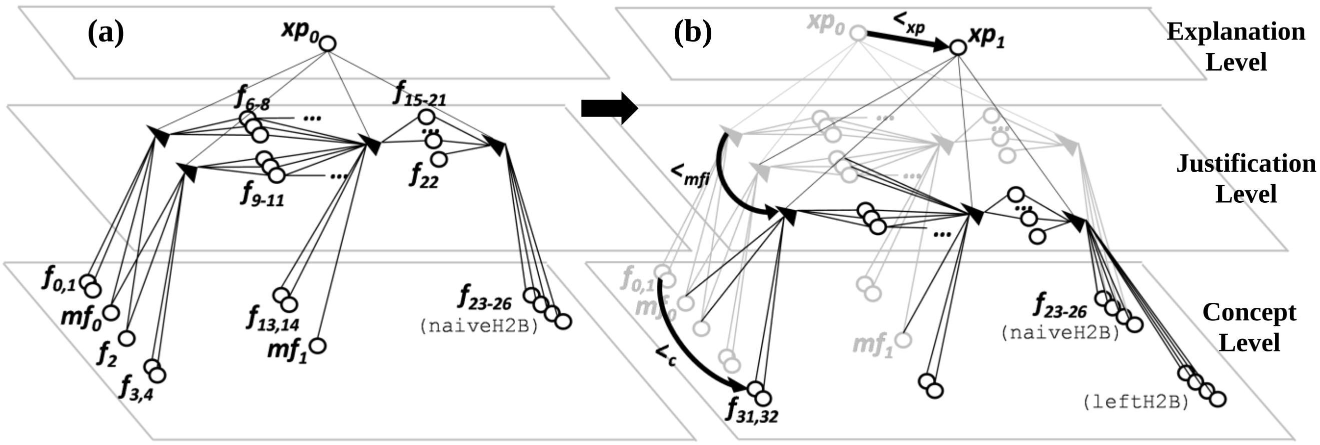
3. Modeling Self-Explanation

In people, knowledge integration requires more than just memorization; otherwise, students who self-explain each sentence in a passage would not have learned more than students who read each sentence twice [Chi *et al*, 1994]. In our system, self-explanation affects the metaknowledge that organizes and prioritizes domain knowledge. The processes involved include model formulation, preference computation, preference-based pruning, and contradiction handling. Suppose the system contains the "single loop" model discussed above, and it is integrating the textbook information from Chi *et al*, "The septum divides the heart lengthwise into two sides… the left side pumps blood to other parts of the body." Our representation of this knowledge includes the following facts:

```
(partitionedInto Heart (LeftRegionFn Heart))
(partitionedInto Heart (RightRegionFn Heart))
(isa l-heart (LeftRegionFn Heart))
```

```
Self-Explain (new-facts M, network N)
    N.DT ∪= M  ;; ∪= ≡ increment via set union
    ComputePrefs(N.DT)
    for each (s in N.DT.Situations)
        newXPs = DoModelFormulation(s, N.DT)
        N.Justifications ∪= newXPs.Justifications
        N.Explanations ∪= newXPs
    ComputePrefs(N.Justifications)
    DetectInconsistencies(N.Explanations)
    ComputePrefs(N.Explanations)
```

Figure 2: High-level self-explanation procedure.



Legend

f_0	(isa heart Heart)	f_{15}	(isa mfi2 FluidFlow)
f_1	(physicallyContains heart Blood)	f_{16}	(fromLocation mfi2 mfi0)
f_2	(isa Blood StuffType)	f_{17}	(toLocation mfi2 mfi1)
f_3	(isa body WholeBody)	…	…
f_4	(physicallyContains body Blood)	f_{22}	(describes mfi1 naiveH2B)
mf_0	ContainedFluid	f_{23}	(isa naiveH2B PhysicalTransfer)
f_5	(greaterThan (Amount Blood heart) 0)	f_{24}	(substanceOf naiveH2B Blood)
f_6	(isa mfi0 ContainedFluid)	f_{25}	(outOf-Container naiveH2B heart)
f_7	(substanceOf mfi0 Blood)	f_{26}	(into-Container naiveH2B body)
f_8	(containerOf mfi0 heart)	…	…
…	…	f_{31}	(isa l-heart (LeftRegionFn heart))
mf_1	FluidFlow	f_{32}	(physicallyContains l-heart Blood)

Figure 3: Tiered network relating explanations (top), justification structure (middle) and conceptual knowledge (bottom). (A): After explaining the heart pumps blood to the body. (B): After explaining the left-heart pumps blood to the body, with preferences across concepts ($<_c$), model fragment instances ($<_{mfi}$), and explanations ($<_{xp}$).

```
(isa leftH2B FluidFlow)
(outOf-Container leftH2B l-heart)
(into-Container leftH2B body)
(substanceOf leftH2B Blood).
```

When new facts are received, the system uses self-explanation to integrate them. Figure 2 summarizes the operations involved. It entails explaining all new situations (e.g. `FluidFlow` instance `leftH2B`), as well as using new propositions and model fragments to re-explain previous situations. The system achieves this by computing concept-level preferences ($<_c$) between entities (e.g. `heart` vs. `l-heart`), performing model formulation to explain new situations (e.g. `leftH2B`) and previous situations affected by new concept-level preferences (e.g. `naiveH2B`), detecting inconsistencies, and computing preferences between the resulting model fragment instances and explanations ($<_{\text{mfi}}$ and $<_{\text{xp}}$, respectively). We discuss each in turn.

3.1 Explanation-Based Knowledge Organization

In our system, domain knowledge is organized in a knowledge-based tiered network as in Friedman and Forbus [2010]. Figure 3(a) shows an explanation of blood flowing from the heart to the body, per the "single loop" model discussed above. Figure 3(b) shows the same network after the system explains the new situation `leftH2B`, re-explains the intuitive situation `naiveH2B` with new facts, and computes preferences between new and prior knowledge. The network contains three tiers:

1. The bottom (concept) tier contains instructional and intuitive facts from the domain theory. This includes facts about entities (e.g. $f_{1\text{-}4}$), model fragment types (e.g. $mf_{0\text{-}1}$), and situations requiring explanation (e.g. $f_{23\text{-}26}$).
2. The middle (justification structure) tier plots intermediate beliefs (nodes) and justifications (triangles) which associate antecedent and consequent beliefs. Justifications with model fragment types as antecedents represent model fragment instantiations.
3. The top (explanation) tier plots explanations (e.g. xp_0). Each explanation represents a set of justifications (triangles) that provide *well-founded support* for some situation to be explained, such that the justification structure is free of cycles and redundancy.

Each explanation node also refers to a logical context where all of the antecedents and consequences of its component justifications are believed. Inconsistency within an explanation context is penalized, as described below, but inconsistency across explanations is permissible.

In addition to justification structure and explanation association, metaknowledge includes inconsistency assertions, information source assertions (e.g. whether a belief was learned via instruction), and epistemic preferences. In the following, we discuss the metareasoning processes that make these assertions.

3.2 Epistemic Preferences

Instead of retracting assertions from the knowledge base, our model uses metaknowledge-based preferences. Preferences provide a means of encoding bias in our learning system, in searching for participant entities during model formulation, and in searching for explanations during question-answering. Preferences are encoded across concepts, model fragment instances, and explanations, as shown in all three tiers of Figure 3(b). We first discuss how preferences are used, and then how they are computed and aggregated during self-explanation.

3.2.1 Using epistemic preferences

Preferences are used for two main tasks: (1) pruning entities during model formulation; and (2) selecting knowledge for reasoning during question-answering after learning.

If a preference exists between two entities (e.g. the left-heart entity `l-heart` is preferred over `heart` in Figure 3(b)), the non-preferred entity will not be considered for participant binding during model formulation. This reduces the number of model fragments instantiated during model formulation, and also ensures that non-preferred concepts do not proliferate into new explanations. These preferences at the concept level are propagated upward to preferences between model fragment instances (e.g. the `ContainedFluid` of `l-heart` is preferred over the `ContainedFluid` of `heart` in Figure 3(b)). These are in turn propagated upward to preferences between entire explanations (e.g. $xp_0 < xp_1$ in Figure 3(b)).

Preferences between explanations are used for later reasoning. For example, $xp_0 < xp_1$ indicates that the constituent model fragments and associated concepts of xp_1 are favored over xp_0 when reasoning about `naiveH2B` or similar phenomena. The non-preferred – and, in this case, overgeneral - explanation xp_0 remains present, in case new information retracts the preference.

3.2.2 Computing epistemic preferences

In **ComputePrefs** (Figure 2) a set of rules is used to compute preferences between concepts, model fragment instances, and explanations, along several dimensions. Due to the multidimensionality of preferences, cycles often arise in the preference graph. The system employs a preference aggregation function to compute an aggregate ordering across all dimensions. We first discuss each dimension of preference, and then describe the aggregation function.

Preferences over concepts are represented by arcs between nodes in the bottom tier of the network (e.g. in Figure 3), and are computed along three dimensions: specificity, instruction, and prior-knowledge.

Specificity. Concept-level specificity preference $c_0 <_{\text{c,s}} c_1$ asserts that concept c_1 (e.g. `l-heart`) is more specific than c_0 (e.g. `heart`), *ceteris paribus*. These are inferred via rules in the domain theory, e.g. a region that is a partition of another is more specific. Our simulation uses both domain general and domain-specific rules for specificity.

Instruction. Concept-level instruction preference $c_0 <_{\text{c,i}} c_1$ asserts that c_1 (e.g. `l-heart`) is supported by instruction, and c_0 (e.g. `heart`) is not. This is only computed between comparable concepts such that $c_0 <_{\text{c,s}} c_1$ or $c_1 <_{\text{c,s}} c_0$.

Prior-knowledge. Concept-level prior-knowledge preference $c_0 <_{\text{c,n}} c_1$ asserts that c_1 (e.g. `heart`) is a prior (pre-instruction) concept, and c_0 (e.g. `l-heart`) is not. Like $<_{c,i}$, these are only computed between comparable concepts such that $c_0 <_{c,s} c_1$ or $c_1 <_{c,s} c_0$.

Preferences over model fragment instances are represented by arcs between model fragment instantiations in the middle tier of the network (e.g. in Figure 3).

Specificity, instruction, and prior-knowledge preferences $mfi_0 <_{mfi,s/i/n} mfi_1$ are inferred when entities e_0 and e_1 occupy identical participant slots of mfi_0 and mfi_1, respectively, and $e_0 <_{c,s/n/i} e_1$. In addition, all other participants of mfi_0 and mfi_1 must be equal or ordered along the same dimension $<_{c,s/n/i}$, in the same direction. For example, the model fragment instance `ContainedFluid` of `Blood` in `l-heart` is specificity-preferred over the `ContainedFluid` of `Blood` in `heart`, due to the concept-level specificity preference.

Completeness. Model fragment instance-level preference $mfi_0 <_{mfi,c} mfi_1$ asserts that mfi_0 contains participants that are assumed (i.e. bound to skolem terms), mfi_1 does not, and the other participants are identical or comparable along $<_{c,s}$, without respect to direction.

Preferences over explanations $xp_0 <_{xp,s/i/n/c} xp_1$ are inferred when xp_0 and xp_1 contain model fragment instances mfi_0 and mfi_1, respectively, such that $mfi_0 <_{mfi,s/i/n/c} mfi_1$, and all other model fragment instances of xp_0 and xp_1 are identical.

3.2.3 Aggregating epistemic preferences

As noted by Doyle [1991], epistemic preferences along several dimensions can be aggregated into a single dimension. Our system aggregates preferences on the concept, model fragment instance, and explanation tiers, using a preference aggregation function. The inputs to the function include preferences on tier *t*, along all dimensions *d*, which comprise partial orderings $<_{t,d}$. The output is a single partial ordering $<_t$. The function uses a *preference ranking* D_t over dimensions at each tier, e.g. D_c = <s, i, n>, D_{mfi} = <c, s, i, n>, D_{xp} = <c, s, i, n>. For each $d_{i=1\ldots|Dt|}$ in D_t, preferences are aggregated $<_t \mathrel{+}= <_{t,d}$, unless it results in a cycle in $<_t$. This results in a partial *aggregate ordering* over elements in *t*. The preference rankings for each tier are parameters to the simulation, which affects preference computation, and consequently affects model formulation and question-answering. As we discuss below, altering preference rankings can drastically affect the outcome of learning.

3.3 Handling Inconsistencies

After the system performs model formulation, it detects inconsistencies within and across explanation contexts (**DetectInconsistencies** in Figure 2). There are two types of inconsistency: (1) local inconsistency, where there are contradictory assertions within an explanation; and (2) global inconsistency, where assertions in two or more explanations are contradictory, and cannot be used jointly in a larger explanation. Locally inconsistent explanations are not permitted in the aggregate explanation ordering. Globally inconsistent explanations are permissible, but not believable simultaneously – i.e. the model fragments they contain might not be active simultaneously. For example, in the "single loop" or "ebb & flow" models, the `FluidFlow` process from `heart` to `body` has the condition `(greaterThan (Pressure heart) (Pressure body))`, but the reverse is true for the `body` to `heart` flow in the same model. These flows may both be believed, but they must not occur simultaneously. This simultaneity injunction is recorded in metaknowledge.

Model	Pre (c)	Pre (p)	Post (c)	Post (p)
No Loop	3	1	2	0
Ebb/Flow	1	1	0	0
Single Loop	3	7	0	0
Single Loop: Lung	1	3	0	2
Double Loop (1)	0	0	4	2
Double Loop (2)	1	0	3	8

Table 1: Pre- and Post-test models for control (c) and prompted (p) groups in Chi et al [1994].

4. Self-Explanation in Students

We simulate the results from Chi *et al.* [1994], which studied the self-explanation effect on 21 eighth-grade students. This included a pretest of their knowledge of the human circulatory system, reading 101 sentences on the subject, and a posttest. The control group (9 students) read each sentence twice, and the experimental group (12 students) was prompted to explain each sentence after reading it. The experimenters found that the prompted group experienced a significant gain in learning relative to the control group, and prompted students who self-explained most frequently achieved the correct "double loop (2)" model on the posttest. Results are summarized in Table 1.

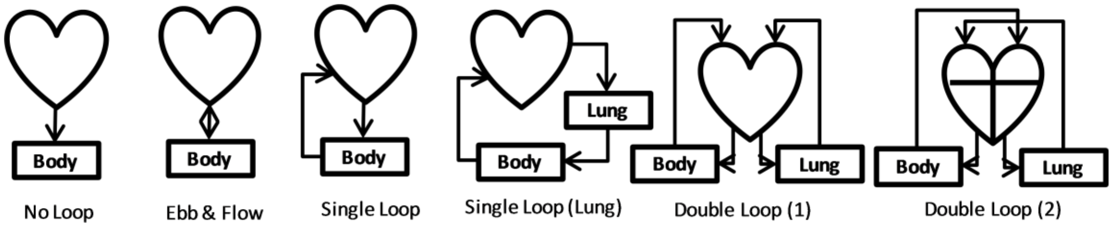


Figure 4: Student models of the human circulatory system from Chi *et al* [1994].

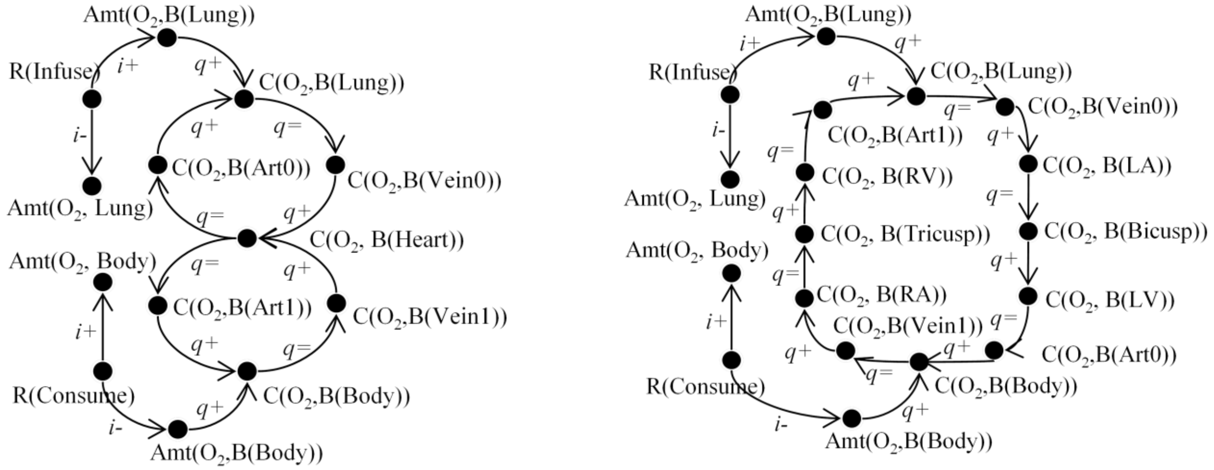


Figure 5: QP Influence graphs generated by the system for Double-Loop (1) (left) & Double-Loop (2) (right). Key: R=Rate; Amt=Amount; C=Concentration; B=Blood-in; (R/L)A=R-/L-Atrium; (R/L)V=R-/L-Ventricle.

5. Simulation

We simulated knowledge integration using a predicate calculus representation of knowledge from the same textbook passage as Chi's study. We conducted several simulation runs, by varying three parameters: (1) the system's starting model across the student models in Figure 4; (2) the preference rankings; and (3) whether or not the system performs model formulation.

We implemented our model in the Companions cognitive architecture [Forbus *et al*, 2009] and encoded selective knowledge from Chi *et al*'s circulatory system passage using an extended OpenCyc ontology. This included knowledge about the structure of the heart (e.g. left and right sides, upper and lower chambers, valves), direction of circulation, and the diffusion and infusion of both oxygen and carbon dioxide. Importantly, portions of the text did not mention specific entities required for complete model formulation (e.g. flow from the lung to the left atrium was mentioned without immediately mentioning the pulmonary vein), so these were omitted from the CycL encoding as well. Consequently, the system created skolem terms for certain participant entities during model formulation to fill these gaps in the text.

For the pretest and posttest, we queried the system to plot all blood flows, blood concentrations, and related influences on all blood-related quantities. Using explanation-level preferences, the system retrieved prior explanation contexts from a case library of all explanations that describe blood flow, and used the union of all preferred explanation contexts to create QP influence graphs. Two QP influence graphs describing oxygen transport in the circulatory system are shown in Figure 5, for "double loop (1)" and "double loop (2)" circulatory models. Similar graphs were created to describe CO_2 transport and blood flow. These graphs provide the all the information necessary to compare the simulation's model to the student models in Figure 4.

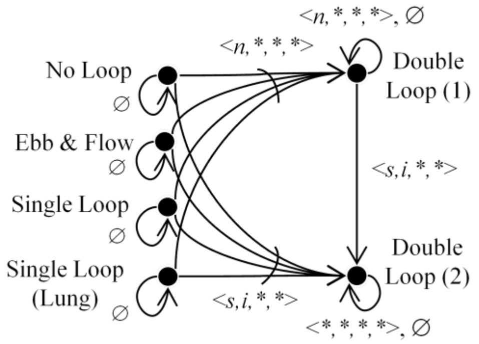


Figure 6: Simulation model transition graph.

Importantly, we cannot expect a single preference ordering to capture the entire control group in Table 1. Students in the control group were not prompted to self-explain, so their individual differences were more apparent. For example, of the three students in the control group who began with the "single loop" model, two of them transitioned to "double loop (1)," and one transitioned to "double loop (2)." Consequently, the system must capture these individual differences with several preference orderings.

As illustrated in Figure 6, by engaging in full self-explanation with preference ranking <*s*,*i*,*.*> (i.e. the last two preferences are irrelevant), the simulation could transition to the proper circulatory model from any initial model. Further, using ranking <*n*,*,*,*> biased the system to use prior (i.e. starting model) concepts (e.g. `heart`) over concepts it learned via instruction alone (e.g. `LeftVentricle`), while doing model formulation. This resulted in the simulation learning the most popular final model in Chi's control group, "double loop (1)" (Figure 5, left), which uses `heart` instead of the more specific concepts used in "double loop (2)" (Figure 5, right). By refraining from any of the self-explanation processes described here (Figure 6, ∅), the system always remained at its initial circulatory model.

Individual differences in the control group were modeled using preference orderings <*n*,*,*,*> (4 students), ∅ (3 students), and <*s*,*i*,*,*> (2 students). The prompted students were modeled using preference ordering <*s*,*i*,*,*> (8 students) and <*n*,*,*,*> (2 students). The remaining two prompted students were not modeled by the system. Both transitioned to the "single loop (lung)" model – one from "no flow" and one from "single loop." The inability of our system to generate these transitions may be due to representation differences, either in the starting knowledge or in the representation of the instructional passage.

By changing starting models and preference rankings, and sometimes ablating self-explanation altogether, the system was able to capture 19 out of 21 (90%) of student model transitions in the psychological data. Individual differences in the control group were captured by three parameter settings, and the majority of the prompted group was modeled by encoding a preference for explanations that contained specific and instructional concepts, <*s*,*i*,*,*>.

6. Discussion & Future Work

We have simulated self-explanation using model formulation, metareasoning, and epistemic preferences. By altering its preference rankings, we are able to affect how the system prioritizes its knowledge and integrates new information. The simulation demonstrates good coverage of the psychological data, as well as a preference ranking <*s*,*i*,*,*> that results in the correct model from any initial model. For autonomous learning systems the preference ranking might need to be more dynamic, reflecting depth of experience versus the credibility of the source.

The task of mapping the flow of blood through the body in a steady state does not require sophisticated temporal reasoning. Consequently, a meta-level temporal representation sufficed for this reasoning task. We have since modified the system to include temporal representations at the object-level. This is required for reasoning about dynamic systems whose component processes are not always active.

While our methods were sufficient to simulate the majority of the students, our model of self-explanation is incomplete. First, people are able to hypothesize system components based on the function of the system. For example, if informed that (1) the lungs oxygenate the blood and that (2) the purpose of the circulatory system is to provide the body with oxygen and nutrients, one might infer that blood flows directly from the lungs to the body. This may have been the case for the two prompted students that were not modeled by the simulation. Second, we believe that there are other processes involved in self-explanation, including spontaneous analogies, rerepresentation, and qualitative simulation. We plan to explore these, along with using natural language encoding of stimuli, in future work.

Finally, we intend to demonstrate the generality of our model by applying it in other knowledge integration domains. Other domains where conceptual knowledge is integrated with intuitive concepts, and for which there exists psychological data, include electricity, evolution, and the changing of the seasons.

Acknowledgments

We thank Bruce Sherin and anonymous reviewers for helpful comments. This work was funded by the Office of Naval Research and Northwestern University Cognitive Science Advanced Graduate Research Fellowship.

References

[Alchourròn *et al*, 1985] C. E. Alchourròn, P. Gärdenfors, and D. Makinson (1985). On the logic of theory change: Partial meet contraction and revision functions. *Journal of Symbolic Logic*, *50*: 510–530.

[Chi *et al*., 1994] Chi, M., de Leeuw, N., Chiu, M., & LaVancher, C. (1994). Eliciting self-explanations improves understanding. *Cognitive Science*, 18: 439-477.

[Chi, 2000] Chi, M. (2000). Self-explaining expository texts: The dual processes of generating inferences and repairing mental models. In R. Glaser (Ed.), *Advances in Instructional Psychology*, Lawrence Erlbaum. 161-238.

[DeJong, 1993] DeJong, G. (*Ed.*) (1993). *Investigating Explanation-Based Learning*. Kluwer Academic Publishers, Norwell, MA, USA.

[Doyle, 1991] Doyle, J. (1991). Rational Belief Revision. *Proceedings of the Second Conference on Principles of Knowledge Representation and Reasoning*: 163-174.

[Falkenhainer & Forbus, 1991] Falkenhainer, B. & Forbus, K. (1991). Compositional modeling: Finding the right model for the job. *Artificial Intelligence*, *51*: 95-143.

[Forbus, 1984] Forbus, K. (1984). Qualitative process theory. *Artificial Intelligence*, *24*: 85-168

[Forbus *et al*., 2009] Forbus, K., Klenk, M., & Hinrichs, T. (2009). Companion cognitive systems: Design goals and lessons learned so far. *IEEE Intelligent Systems*, *24*(4): 36-46.

[Friedman and Forbus, 2010] Friedman, S., & Forbus, K. (2010). An integrated systems approach to explanation-based conceptual change. *Proceedings of the 25th Annual AAAI Conference on Artificial Intelligence*.

[Laird *et al.*,1987] Laird, J., Newell, A., & Rosenbloom, P. (1987). SOAR: An architecture for general intelligence. *Artificial Intelligence*, 33(1): 1-64.

[Katsuno and Mendelzon, 1991] Katsuno, H., & Mendelzon, A. (1991). Propositional knowledge base revision and minimal change. *Artificial Intelligence, 52*: 263-294.

[Rickel and Porter, 1997] Rickel, J. and Porter, B. Automated modeling of complex systems to answer prediction questions. *Artificial Intelligence,* 93(1-2), 201-260.

[VanLehn *et al.*, 1992] VanLehn, K., Jones, R., & Chi, M. (1992). A Model of the Self-Explanation Effect. *Journal of the Learning Sciences*, 2(1): 1-59.

[Winston & Rao, 1990] Winston, P., Rao, S. (1990). Repairing learned knowledge using experience. *Massachusetts Institute of Technology, AI Memo #1231, May 1990.*