

Towards Applying Case-Based Reasoning to Composable Behavior Modeling

Joshua D. Summers

Dept. of Mechanical Engineering,
Clemson University
Clemson, SC 29634-0921
joshua.summers@clemson.edu

Bruce M. McLaren

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

David W. Aha

Navy Center for Applied Research in Artificial Intelligence
Naval Research Laboratory, Code 5515
Washington, DC 20375
aha@aic.nrl.navy.mil

ABSTRACT: *Modeling complex doctrinal behaviors to support military simulations has recently been extended to graphical interfaces, giving military subject matter experts the ability to create behaviors that can be automatically encoded. For example, the Composable Behavior Technology (CBT) system supports this modeling paradigm. We discuss how to extend CBT to capitalize on previously stored and validated behavior models, and how to apply case-based reasoning to this design problem. We propose an approach that incorporates text and graphical retrieval strategies and multiple levels of similarity analysis. Finally, we outline future research and development directions.*

1. Introduction

Experts who design models of composite behaviors for simulations of computer-generated forces (CGF) appear to rely on their previous experiences; they often adapt existing behaviors to generate new ones. This suggests that case-based reasoning (CBR), a general problem-solving methodology for reusing previously stored solutions [1], may be suitable for managing this activity. While, traditional CBR approaches have been used to retrieve and manipulate structured representations using a pre-determined vocabulary and methods for reasoning about implicit relationships among the vocabulary items, the application of CBR to this problem, including the representations for CBR sub-processes, is not obvious. This requires analyzing several challenging design decisions, including those concerning:

- (1) desired functionality,
- (2) choice of case content,
- (3) case representation,
- (4) case retrieval,
- (5) solution reuse,
- (6) solution revision, and
- (7) case base maintenance.

Challenges (1) and (2) define our system specification and focus, while challenges (3)-(7) define some primary foci of the CBR problem-solving cycle [1]. We briefly discuss each below in the context of developing an assistant for helping users to generate models of CGF behaviors. This paper begins by introducing computer-generated forces and relevant background, followed by a discussion on each of the challenges including potential solutions and directions. We conclude with a discussion of potential similarity metrics (instance, structural, relational, and attribute).

The work presented in this paper is a result of joint collaboration with SAIC in the investigation and development of CGF authoring tools. We were charged with investigating the research issues associated with CBR and with development of possible design solutions. In a companion paper [2], SAIC discusses the implementation of a prototype authoring system building on the theoretical work discussed in this paper.

2. Background

This section provides a brief background on three topics of interest in this paper and investigation: computer-generated forces, composable behavior technology, and

case based reasoning. This is intended as an introductory, non-exhaustive overview of these topics.

2.1 Computer-Generated Forces

Computer-generated forces support simulation of a complex battlefield environment without requiring many human operators. A basic goal of a CGF system is to support behavior modeling that is indistinguishable from human behavior specified by military doctrine. Modeling human behavior in the form of CGF has traditionally been a formidable knowledge engineering task involving the transformation of military doctrine into software. Knowledge acquisition for developing CGF tools is facilitated through subject matter experts (SMEs) using text documents and spreadsheets. SMEs create descriptions of situations, scenarios, behaviors, and actions, typically guided by a structure such as a finite state machine. Software developers then convert these descriptions into behavior programs.

CGF simulations developed in this way with traditional programming languages become difficult to maintain and expand as the number and complexity of behaviors grows. Many behaviors may have common elements, but it is difficult to reuse them when they are embedded in a complex behavior. Different SMEs and software engineers may use different styles and conventions, making it difficult to identify similar elements.

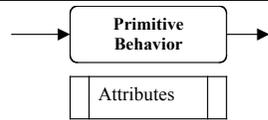
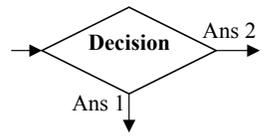
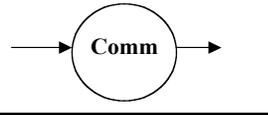
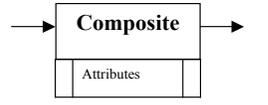
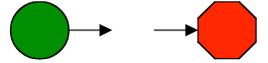
2.2 Composable Behavior Technology

CGF projects have begun to use a composable behavior approach that addresses the scalability problem. In this approach, simple *primitive* behaviors are identified early in development and are coded as reusable behavior components. A primitive behavior is an action that is not decomposable into sub-actions and that is used, combined with other primitive behaviors, to develop *composite* behaviors. The Composable Behavioral Technologies (CBT) project sought to extend this concept in several ways to simplify the creation of complex behaviors from military doctrine (e.g., using an abstraction hierarchy of behaviors) [3, 4]. In CBT, users compose behaviors from primitives or (other) composite behaviors, and store them to encourage reuse. CBT represents the behaviors as objects, representing them graphically, thereby allowing a user to compose behaviors by “dragging and dropping” in the interface. CBT supports several basic elements, as shown in Table 1.

Composite behaviors are instances where attribute (parameter) values are fully specified; there are no defaulted or derived values. These are not currently templates. Thus, the full potential for reuse is not

currently realized in this format, such as through frames from the AI community [5].

Table 1: CBT Primitive Elements

Iconic Representation	Description
	Primitive Behavior: Lowest level of behavior instruction.
	Predicates: Decision point with alternative paths depending on the current situation of the simulation.
	Communications: Orders, either received or sent from/to Superior, Subordinate, or to a specific Role Block.
Arcs with head/tails depending on timing intent (Proposed – in development)	Temporal Constraints: Connections between the different element nodes.
(NOT REPRESENTED)	Role Blocks: Not explicitly illustrated in the current form, they are used to distinguish specific behavior actions for different levels or groups.
	Composite Behaviors: Black box representations of previously defined behaviors for reuse.
	Terminators: These indicate the initial and final nodes in the behavior network.

Logic checkers have been implemented to ensure syntactic validity of the behavior networks. These validity checks include: node, edge, branch, conditional constraint, timing, attribute, adjacency, and network checking. Semantic validity is not performed; this is a challenging and open research problem. The SMEs, who will be the primary users of CBT, are expected to create semantically correct behavior models.

An example of a composite behavior is illustrated in textual form in Figure 1. This behavior represents an “attack by fire” action. The textual description of this behavior includes an annotation option for the author (or editor). For example, an annotation appears at the bottom of Figure 1.

This model is composed of seven numbered elements. The second number in each element points to the next element, defining the temporal sequence of events. The first element listed is “BEGIN”, a terminator; it is followed by the seventh element, “OccupyOWPos”,

which has five attributes associated with it and is related to the sixth element, a composite behavior.

```

52 jackie 26 AttackByFire Thu May 18 14:42:22 2000
EDT 1524 0
(-1 -1 (CONSTRAINT-PAIR (CONSTRAINT-NULL)
(CONSTRAINT-NULL) ) COMPOSITE AttackByFire
(ATTRIBUTES
(Attribute: TOP_LEVEL_ROLE true Attack)
(Attribute: ROLE true ( )))
(SYSTEM-ATTRIBUTES) \
(STATEMENTS \
(1 7 (CON-PAIR(0)(0)) BEGIN ( ATTRIBUTES ))) \
(2 -1 (CON-PAIR (0) (0) ) COMPLETE ( ATTRIBUTES
)) \
(3 -1 (CON-PAIR (0) (0) ) PREDICATE Target? (
ATTRIBUTES) \ ((no 4) (yes 5) ) )
(4 2 (CONST-PAIR (0) (0) ) COMPOSITE
PltMachineGunFireAtLoc ( ATTRIBUTES
(Attribute: Saved-ID true 96) ) NOT-
ELABORATED) \
(5 2 (CONST-PAIR (0) (0) ) COMPOSITE
PltFireAtTarget ( ATTRIBUTES (Attribute: Saved-
ID true 117) ) NOT-ELABORATED) \
(6 3 (CONSTRAINT-PAIR (NULL) (NULL) )
COMPOSITE PltScanForThreat ( ATTRIBUTES
(Attribute: Saved-ID true 83) ) NOT-
ELABORATED) \
(7 6 (CONSTRAINT-PAIR (NULL) (NULL) )
PRIMITIVE OccupyOWPos ( ATTRIBUTES
(Attribute: DismountedSpeed true Specific 1.5
MIN: 0.0 MAX: 3.4028235E38) (Attribute: Route
false MissionAttribute Route) (Attribute: Speed
true Specific 30.0 MIN: 0.0 MAX:
3.4028235E38) (Attribute: SpeedLimit true
Specific 0.0 MIN: 0.0 MAX: 3.4028235E38)
(Attribute: Formation true Specific Formation
Line)) \
(ROLE-STATEMENTS ) )
Mon Jan 08 12:55:25 2001 EST
act2 3 12
Platoon level attack by fire. The platoon will determine
the enemy type and deliver the appropriate fire with
the appropriate weapons accordingly.

```

Figure 1: Adapted Text of the “Attack By Fire” Composite

Some elements do not have succeeding events, such as the second (“COMPLETE”) and the third (“PREDICATE Target?”). These elements have a number designation of (-1) to indicate that there are no subsequent events. The heading of the behavior model includes database entry, author, behavior name, and creation date. Other information is included to encode the behavior model.

A graphical representation of this behavior is shown in Figure 2. This representation, using the iconic vocabulary described in Table 1, includes five types of elements: terminators (BEGIN and COMPLETE), primitive behaviors (“PltScanForThreat”, “PltFireAtTarget”, and “PltMachineGunFireAtLoc”), a decision node (“Target?”), and a composite behavior

(“OccupyOWPos”). The relationships, represented by arrows between the nodes, are not specified with respect to temporal intent.

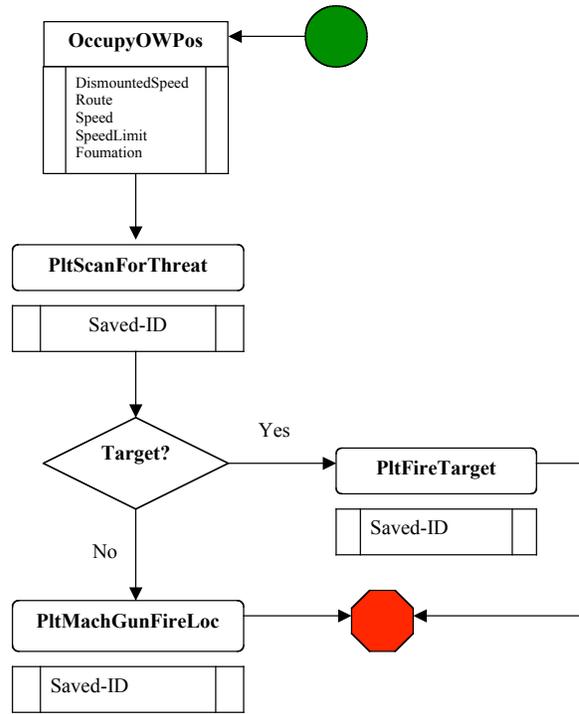


Figure 2: Graphic of the “Attack By Fire” Composite.

Several composite behaviors have been developed (e.g., one is labeled Alpha Main Gun Fire and is described as “alpha tanks fire their main guns at the recommended target”, while another is labeled Shoot 4-25 Location A and is described as “shoot bursts of 25 rounds at each of the enemy locations”). Additional behaviors are being developed by USMC SMEs.

2.3 Case-Based Design

CBR systems can help designers in various design stages by (1) providing them access to a library of valid designs that satisfy, or nearly satisfy, the defined requirements, functions, or characteristics and (2) modifying the retrieved design cases to satisfy these requirements. Example CBR systems include KRITIK [6], HICAP [7], SIROCCO [8], and FABEL [9]. The CBR process typically involves four tasks. First, *retrieval* finds past design cases that may be appropriate in the new design scenario and validates them with respect to the new design requirements. Second, *reuse* adapts and combines the cases to fully satisfy the new problem’s design requirements. Third, *revision* involves evaluating, and possibly repairing, the new composite. Finally, *maintenance/retention* updates the case base (e.g.,

integrating this new composite). CBR systems often have four elements: a case base (which stores previous designs for retrieval), a case retrieval engine (which allows designers to express retrieval queries), an adaptation module (which encapsulates design knowledge to partition or combine retrieved cases), and a validation module (which is used to help evaluate and ensure the validity of combining previous designs when generating a new design).

Our goal is to develop, evaluate, and provide an interactive authoring tool for composite behavior modelers that will use case retrieval, adaptation, and management (e.g., retention/learning) algorithms. For example, we anticipate that users will (incrementally) specify the desired functionality of a new composite behavior, the system will respond with stored behaviors that have some relevance to it, and the user will adapt a retrieved behavior and/or compose multiple behaviors to construct the new behavior. Thus, authoring can be viewed as a case-based design (CBD) task [10]; the objective is to synthesize a new behavior by composing existing primitive behaviors using available composition operators supported by the CBT tool [4]. CBD is generally an iterative process that combines deductive, inductive, and abductive reasoning, possibly switching between different representations and abstraction levels. Design problems, whose tasks include synthesis, analysis, evaluation, and representation mapping, consist of the desired goals and target values, known constraints and relations, and user controlled independent variable values, while the design artifact represents the final solution that satisfies the specified design problem at some level of satisfaction (i.e., feasible, satisficing, robust, or optimal) [11]. From this perspective, the design *problem* is the creation of a desired behavior to model while the design *artifact* is a final composite behavior. Typically, designers start from a previous solution, possibly using the past design's rationale, to design a new artifact. Design can be classified as *generative* or *variant*. In generative design, the designer creates from scratch the solution. In variant design, the designer modifies existing solutions to satisfy the new problem.

Maher *et al.* [12] identify three major issues associated with case representation for CBD: design case content, representation paradigms for case memory organization, and user view of the cases. Design case content is the design data and documents constituting the case. It can include strings of parameters and values, collections of design tasks undertaken to achieve the design solution, collections of all design documents generated in the process, etc. The content of design cases must be processed (manually or automatically) to capture their characteristics and properties. Traditionally, this

processing is done *a priori*. These characteristics and properties are added to cases through annotations. Storage and performance issues in memory organization are addressed by providing the best organization of design cases, usually with respect to their similarities. Finally, users view and access the cases based upon their own understanding of the domain and the specifications of their design problem. Some tools exist for authoring cases interactively, such as CASCADE [13], CBR Works [14], or k-Commerce [15].

One of the primary goals of indexing cases in a case base is to achieve good performance without sacrificing quality of retrieval. The indexing problem poses some inherent trade-offs between the desire to provide a flexible problem definition approach and the desire for using an efficient retrieval algorithm. As the size and complexity of the case base grows, an indexing scheme is required to provide quick access to the cases. However, the scheme might provide only a limited view of each of the design cases, a view that is not always entirely compatible with new problems. The view of the cases is limited by the characteristics that were selected in the development of the index. Indices should be chosen such that they aid in predicting the solution, that they be sufficiently abstract to span several different cases, and that they be sufficiently concrete to be applicable in future problems [16]. Much of the effort in developing indices for case bases focuses on domain specific classification and organization. Most of the CBD approaches rely on indexing design data with respect to annotations capturing their attributes. Selecting the properties and characteristics to represent design data has significant consequences. Indeed, in existing systems selected attributes constrain the level of expression in the retrieval phase. Different techniques and approaches are used to perform indexing [17]. These approaches enable the indexing of the entire case base, yet do not offer additional granularity (flexible levels of abstraction).

Some examples of query interfaces that rely heavily on indexed systems include expert driven questionnaires, tabular, frame based, textual, or function structures [18]. Retrieval queries usually are Boolean expressions of terms consisting of valued attributes as represented through the indices (or pre-computed annotations). Generally, it is unlikely to find an exact match for the design [19]. Portions of the knowledge base set of designs are needed to generate a design that will satisfy the design problem. Therefore, a system should be capable of retrieving portions of designs based on matching pieces of the higher-level signatures. The dependencies may include: intersecting features, tolerances of one feature based on data of a second feature, features in close proximity, same tool used in two features, or same approach direction used in two features. This approach is used to index slices,

which may be retrieved by the system if the slice signatures can be found in the original design signature.

3. Research Issues

This section revisits the issues identified earlier that are of specific interest to designing a CBR system to support CBT behavioral modeling. These issues include: desired functionality, choice of case content, case representation, case retrieval, solution reuse, solution revision, and case base maintenance. Each issue is explored with respect to required research.

3.1 Desired Functionality

The existing tool, CBT, is a generative design tool, providing the behavior modeler with the ability to define new composite behaviors. We propose to use CBD to extend CBT to support variant design. In particular, we view the CBD task as incrementally acquiring a problem description (i.e., a query) from the user, and providing insight on the existing behaviors and methods for composing them that could be used to satisfy the user's needs. Commercialized interactive tools have been deployed to support diagnosis tasks (e.g., help-desk assistants), but not for design tasks. Furthermore, we anticipate that some mixed-initiative interaction capabilities (i.e., where the system learns to take and share the initiative in helping the human) should be supported in this system; this is an ongoing area of interest to the CBR community [20, 21]. ***Thus, our challenge is to investigate mixed-initiative approaches for CBD.***

3.2 Choice of Case Content

CBT composite behaviors are modeled in different modes: graphical (diagrams of the behavior flow), textual (short descriptions of the behavior), and mixed (attributes of the elements). We anticipate that cases will be in the form of <problem, solution> pairs, where problems are in a mixed relational format derived from the text descriptions, and solutions are in graphical format (i.e., each element is linked to other elements (arcs) through relations (edges), creating a directed graph, where each arc and edge is annotated with appropriate information). ***This diversity of content is an issue that must be addressed in an integrated manner, and reasoning with such diversity poses a challenge for an integrated CBR strategy.***

3.3 Case Representation

This concerns the representation of the queries and cases to the user, and this choice will impact our system's interaction functionality. Several CBR systems have been developed that work with structured representations

of case solutions that can be used to capture graph-represented content (e.g., [8, 22, 23]), and commercial tools exist that implement algorithms to support object-oriented representations (e.g., empolis' e:kbs system and orange architecture). In our situation, we will also need to work with text descriptions for summarizing these behaviors. Several advances in Textual CBR [24] have described methods for reasoning with cases whose contents are in text form. Likewise, the composite modeling problem requires reasoning with this text (i.e., associating it with specific aspects of the stored behavior's functionality). This requires semantic interpretation of text data, which has been the focus of an ongoing Naval Research Laboratory (NRL) project [25]. NRL has developed expressive representations for text in linguistic ontologies using an extension of Generative Lexicon (GL) theory [26], and we believe that this approach for generating these simple ontologies could be used for our objective. ***Integration of graphical case solutions with our GL-inspired text interpretation process poses a technical challenge.***

3.4 Case Retrieval

Traditional approaches to incremental query formation include using concise natural language text strings, feature identification and attribute value assignment (e.g., thru a *conversational* CBR interface), and/or partial case modeling (e.g., using modes in which users can identify query components by interacting with diagrams, scripts, or interfaces (e.g., [10]). After parsing, cases are returned to the user, typically in a ranked order based on a predefined set of metrics. Similarity metrics for supporting the retrieval of composite behaviors will operate primarily on the case problem (i.e., interpreted text descriptions in a relational format). Retrieval of cases should include syntactic and semantic similarity (reusability) measures. Semantic similarity measures are typically domain specific and driven by the defined ontology [9]. While many authors have described how to learn similarity knowledge (e.g., using feature weights [27]), we expect that our focus on retrieval will be on using ontologies to assist with computing semantic similarity. Our challenge is to develop such ontologies and machine learning algorithms for revising them through explicit and implicit user feedback. ***Few previous efforts have focused on the automatic updating of ontologies.***

3.5 Solution Reuse

Solution reuse in CBR includes automated adaptation of the retrieved and selected cases to satisfy the specified problem query. Although some work exists on inducing adaptation knowledge in a domain-independent manner (e.g., [28, 29]), previous efforts have not addressed issues

concerning graph-structured solutions. In the case of composite behavior modeling, adaptation rules may be required for air, land, or sea operations. *Acquiring the necessary domain-specific case adaptation knowledge and maintaining this knowledge independent of the reasoning system is an open issue.*

3.6 Solution Revision

This includes the interactive revision of the retrieved and selected cases by the human user, which permits user control of adaptation. Flemming et al. [30] discuss an initial approach to case adaptation that allows the user to interactively refine the retrieved case or to automatically adapt the case based on the “static” adaptation domain specific knowledge. This approach to refinement is limited to a predefined structure of the solutions and is limited to value modification. *A more robust revision approach is needed in the composite behavior-modeling problem that provides users with the flexibility to modify, delete, or add information to the retrieved cases at multiple abstraction levels.*

3.7 Case Base Maintenance

As new cases (i.e., behavior models) are created, they need to be integrated with the case base for future retrieval and adaptation. Case maintenance must address indexing (dynamic vs. static), redundancy elimination (storing only unique cases vs. storing all cases), and consistency checking. The nature of this domain of behavior modeling dictates that a rigid vocabulary known *a priori* is not feasible. For this reason, strict structuring of the cases and the knowledge within the cases is difficult at best. *These are still open research issues for unstructured and mixed representation cases such as those found in CBT’s composite behavior models.*

4. Proposed Directions and Rationale

Our proposed architecture for CBD enhanced composable behavior modeling is roughly split into two sub-systems: lexical and graph retrieval. These two sub-systems are combined with a set of similarity agents and case bases of primitive and composite behaviors. The user interacts with the system via lexical or graphical “conversation”. Should the user decide to proceed in a lexical manner, the traditional conversational CBR (CCBR) approach [10] will be used and a conversation with the lexical CCBR (LCCBR) agent proceeds. The LCCBR agent coordinates between the lexical indexing agent and the similarity agents. Refinements to the user query are proposed to the user while potential cases are presented through the similarity agents. The graphical retrieval module is similar to the lexical module, where the distinction is in the type of representation used: graphical rather than

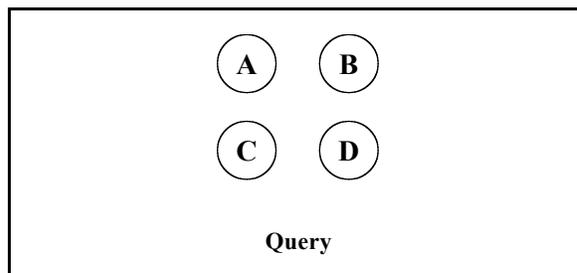
lexical. As the similarity agents compile lists of potential retrieval cases, the list of cases is reconfigured into two views: lexical and graphical. As the user develops a new composite, it is entered into the case base through the respective indexing agents. Architecture extensions can be added to include adaptation agents. Currently, user interaction is the mode for adaptation.

We identified many research challenges above. Here, we expand on and illustrate a core aspect of research on similarity analysis. This is not intended to be a final solution, but rather an initial guideline for directing development. Similarity measures have been widely recognized as key to case retrieval [31, 32]. For the problem domain of composable behavior modeling, we suggest four similarity metrics, namely object, structural, relational, and attribute similarity, some of which may be used in combination.

4.1 Object Similarity

The graphical CCBR agent needs to generate queries as users begin to build behavior models. After a threshold has been exceeded (to be determined in future research, but likely greater than two behaviors), a retrieval session will be initiated. First, a filter on all available composite behaviors will be applied that eliminates those composites that do not include at least one instance of each of the behaviors/composites that have been specified by the user. This includes searching hierarchical descendants found through expanding composite behaviors to the base level. The object similarity measure will initially be defined as the number of elements that are shared between the query and the case. Figure 3 illustrates a sample query and three cases.

Elements of type A, B, and C are found in Case 1. Therefore, this case has a similarity of 3/4 (75% match). Likewise, Case 2 has 3 out of 4 elements in common with the query (A, C, and D). Some ambiguity remains with respect to Case 3, where there are two instances of the element A. This ambiguity requires domain specific knowledge that can be derived from experimentation and interrogation of SMEs. To differentiate between the similarity of Cases 1 and 2, weightings could be used depending on the preference of the user or query author.



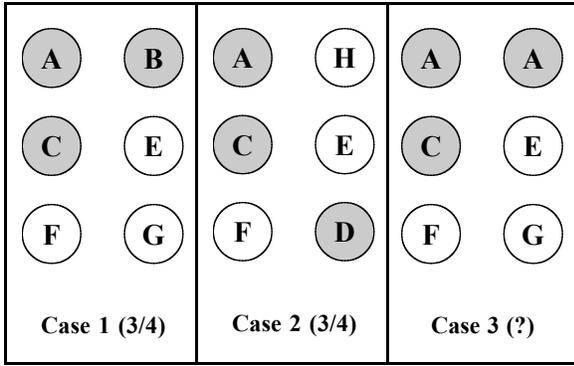


Figure 3: Object Similarity Examples

The object similarity measure may also include matching for inherited graph nodes. Currently, primitive behaviors in CBT are not represented in inheritable form, as each instance of a primitive behavior must be fully defined with respect to the attribute values. An object similarity approach could also be used as an initial “filter” for a more expensive structural similarity method [8]. Using two phases, a retrieval mechanism might first apply relatively inexpensive, but somewhat imprecise, object similarity and then focus on the top candidates from this phase in a more expensive structural evaluation, discussed next.

4.2 Structural Similarity

Structural similarity measures can be developed to evaluate and compare composites and initial queries based on their sub-graph isomorphism. The sub-graph isomorphism problem is known to be NP-Complete [33]. Thus, the complexity of the case graphs will increase the complexity of the similarity measurements. A trade-off between efficiency and accuracy is found when introducing this measure to the proposed behavior retrieval tool. Narrowing the set of cases that are structurally analyzed (e.g., using an initial object similarity filter) helps address the complexity problem, at least in terms of the reducing the cases that are considered and analyzed [7]. However, addressing the isomorphic problem per case is still a challenge.

To illustrate the isomorphic problem recast as a similarity problem, consider the graph in Figure 4, which illustrates five potential behaviors as defined in a graphical query. These behaviors are all found in the subsequent two graphs representing two cases (Figure 5 and Figure 6). If these behavior models have been properly developed (an important assumption in CBR is that the cases are all acceptable), then the similarity between the first (query graph) and the other graphs can be derived.

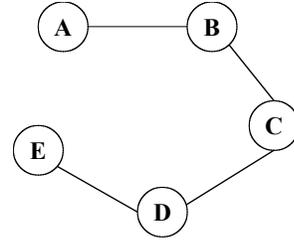


Figure 4: Query

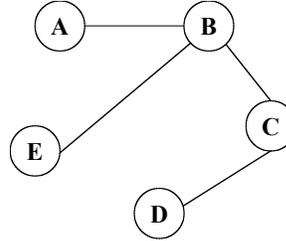


Figure 5: Case 1 (C1)

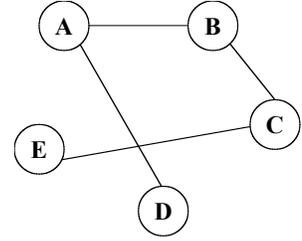


Figure 6: Case 2 (C2)

An initial similarity approach compares the distances between elements in the graph. Table 2 illustrates the distances between each element for the three graphs. Based on this simple approach, Case 1 is found to be more similar than Case 2. The complexity of this approach is $O(n!)$ per case, where n is the number of elements to be matched.

Table 2: Structural Similarity Computations

	Query	C1 Dist	C1 Diff	C2 Dist	C2 Diff
AB	1	1	0	1	0
AC	2	2	0	2	0
AD	3	3	0	1	2
AE	4	2	2	3	1
BC	1	1	0	1	0
BD	2	2	0	2	0
BE	3	1	2	2	1
CD	1	1	0	3	2
CE	2	2	0	1	1
DE	1	3	2	4	3
Total			6		10

One possible heuristic that might reduce the complexity of examining the entire structure of the composite behavior case would be to recognize that decision points indicate a potential break for sub-graph division. Behaviors found before a decision point are separated from those found after a decision point as the flow depends on the result of the decision. Therefore, pre-processing each case into sub-graphs divided at the decision point nodes could significantly reduce the overall complexity of the structural similarity measure.

This metric considers only that a relationship between elements is present, not whether the relationship is directed. In this manner, a directed case graph, as in the case of behavior models (especially those that include temporal relations), requires more specialized similarity measures. This metric may be augmented with consideration paid to the types of relationships used in the models.

4.3 Relational Similarity

In addition to structural similarity, these graph nodes are connected through multiple types of relationships that imply precedence order. For this reason, structural similarity can be augmented with additional information. The direction (precedence order) of the relationships between behaviors may be important in some situations (e.g., it is important to first load a gun and then fire it). However, the precedence between finding a target and then loading a weapon may be irrelevant (depending on the weapon). Additionally, temporal constraints may be used to refine the similarity measure. McLaren's SIROCCO model, for instance, leveraged relational similarity such as domain-specific actions (e.g., an engineering firm *hires* an engineer, an engineer *writes* a report) and temporal order to render structural similarity more tractable [8].

4.4 Attribute Similarity

In addition to the features (entities) and structure (relations), similarities measures can be derived that are based on comparisons between the values assigned to the attributes of the matched instances. These attribute values may include speed of movement, altitude, etc. It is not clear how to weight the attribute similarities (i.e., is being within 10% of the altitude the same as being within 10% of the distance?). Several aggregate functions have been proposed in the literature for synthesizing a global similarity measure [23] (i.e., weighted average, Minkowski, maximum, minimum, k-maximum, or k-minimum). Weighting is used to compare attributes of different significance, as defined by the user or defined within the domain. These similarity metrics must be explored through the guidance of a SME and completed system assessment.

5. Hypothetical Session

To illustrate the envisioned system, we describe a hypothetical session in this section. Two CCBR approaches are illustrated: textual (Figure 7) and graphical (Figure 8). Each approach provides different levels of detail, different types of information, and therefore

different value to the user. Our goal is to integrate both of the approaches with CBT to support variant design of the composable behavior models.

The textual approach is predicated on the availability of suitable descriptions for the composite behaviors. The retrieval interface is divided into question and case solution sections. The questions are generated from the indexed set of cases as a filtering mechanism to refine case selection. Cases are ranked according to a set of similarity measures. Likewise, two sections are illustrated for the graphical approach: the "question" section and the "case" section. The former presents sub-graphs of the behavior models that may be used to refine the search in the indexed set of behavior models. In presenting information in this format, we hypothesize that designers will be able to quickly assess whether the sub-patterns match the given problem.

6. Conclusion and Future Work

We identified several research issues at the intersection of CBR with composite behavior modeling, including:

- (1) investigate mixed-initiative approaches for CBD;
- (2) integrate and reason with a diversity of content;
- (3) supplement generative lexicons with graphical representations;
- (4) automate ontology updating;
- (5) derive and apply adaptation (semantic similarity) knowledge;
- (6) support true interactive revision (user driven) and reuse (system driven) approaches; and
- (7) investigate dynamic vs. non-indexing approaches for mixed representations.

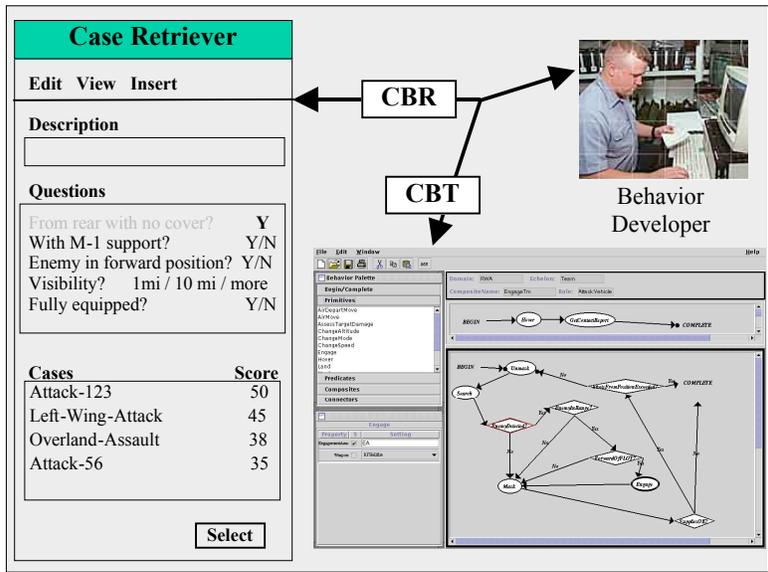


Figure 7: Text Retrieval Interface

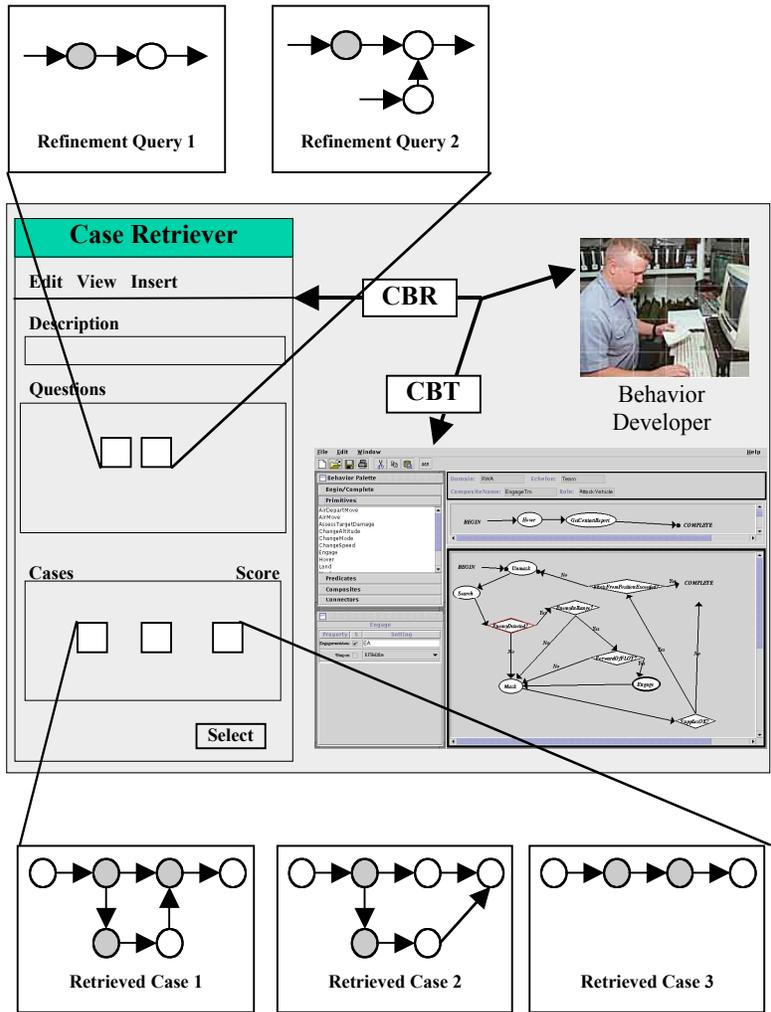


Figure 8: Graph Retrieval Interface

These issues form the foundation for on-going research in supporting composite behavior design through CBD retrieval strategies. This paper highlights areas of research that have been identified in our investigation of the potential application of CBR to composable behavior modeling.

We presented an approach for augmenting CBT behavior modeling using a CBR method. To fully realize the potential of this integration, an initial step is to canonically derive a vocabulary of primitive behaviors, decision nodes, and commands. The current vocabulary appears to be sufficient for specific domain applications, but requires significant effort to extend as new domains are introduced. While logical checking has been incorporated into CBT, semantic validation might also be included as a finite vocabulary is developed. One example of semantic validation could be to determine if necessary resources are available for a given behavior. Consider the sequence of behaviors: “aim weapon”, “load weapon”, and “fire weapon”. The ordering of these behaviors is important because a weapon must be loaded before it can be fired. This precedence between behaviors is implicitly found within properly constructed cases. However, explicit constraints might be used to validate the semantics of the behavior models. These issues, and others, must be addressed in the development of the proposed system.

Acknowledgements: Many thanks to Doug Reece, Jenifer McCormack, and Jackie Zhang, who introduced us to this topic, collaborated with us earlier on it, and have a companion paper in this conference that includes discussion of SAIC’s ongoing implementation efforts.

References

- [1] Aamodt, A., & Plaza, E. (1994). Case-based reasoning: Foundational issues, methodological variations, and system approaches. *AI Communications*, 7, 39-59.
- [2] Reece, D., McCormack, J., & Zhang, J. (2004). A Case-Based Reasoning Tool for Composing Behaviors for Computer Generated Forces. *Proceedings of 2004 Conference on Behavior Representation in Modeling and Simulation (BRIMS)* Arlington, VA.
- [3] Lippe, S., & Courtemanche, A. (1999). Advances in the execution engine for composable behaviors. *Proceedings of the 8th Conference on Computer Generated Forces and Behavioral Representations*. Orlando, FL.
- [4] McCormack, J., von der Lippe, S., & Kalphat, M. (2000). Embracing temporal relations in composable behavior technologies. *Proceedings of the 9th Conf. on Computer Generated Forces and Behavior Representation*. Orlando, FL: IEEE Press.
- [5] Minsky, A., (1985), “A Framework for representing knowledge”, *Readings in Knowledge Representation*, R. Brachman, H. Levesque (Eds.), Morgan Kaufmann Publishing.
- [6] Goel, A., & Chandrasekaran, B. (1989). Functional representation of designs and redesign problem solving. *Proceedings of the 11th International Joint Conference on Artificial Intelligence* (pp. 1388-1394). Detroit, Michigan: Morgan Kaufmann.
- [7] Muñoz-Avila, H., McFarlane, D., Aha, D.W., Ballas, J., Breslow, L., & Nau, D. (1999). Using guidelines to constrain interactive case-based HTN planning. *Proceedings of the 3rd International Conference on CBR* (pp. 288-302). Seon, Germany: Springer.
- [8] McLaren, B.M. (2003). Extensionally defining principles and cases in ethics: An AI model. *Artificial Intelligence*, 150, 145-81.
- [9] Gebhardt, F., Voss, A., Grather, W., & Schmidt-Belz, B. (1997). *Reasoning with complex cases*, Norwell, MA: Kluwer Academic Publishers.
- [10] Summers, J., Lacroix, Z., & Shah, J. (2002). Case-based design facilitated by the design exemplar, *Proceedings of the 7th International Conference on Artificial Intelligence in Design* (pp. 453-476). Cambridge, UK.
- [11] Summers, J., & Shah, J. (2003). Developing measures of complexity for engineering design. *ASME DETC-2003*, DTM-48633, Chicago, IL.
- [12] Maher, M., Balachandran, M., & Zhang, D., (1995), *Case-Based Reasoning in Design*, Lawrence Erlbaum, Mahwah, NJ.
- [13] McKenna, E., & Smith, B., (2001). An interactive visualization tool for case-based reasoners, *Applied Intelligence*, 14, 95-114.
- [14] Empolis Knowledge Management, (2001), “Products: CBR Works”, <http://www.km.empolis.com/start.htm>, September 26, 2001.
- [15] Inference Corporation, (2001), “Inference Corporation – eGain”, [url](http://knowledge.egain.com/), <http://knowledge.egain.com/>, September 26, 2001.
- [16] Kolodner, J., (1991), “Improving Human Decision Making Through Case-Based Decision Aiding”, *AI Magazine*, 12(2), pp. 52-68.
- [17] Bergmann, R., (2002), *Experience Management: Foundations, Development Methodology, and Internet-based Applications*. Berlin: Springer.
- [18] Aha, D., Breslow, L., & Muñoz-Avila, H. (2001), “Conversational Case-Based Reasoning”, *Applied Intelligence*, 14(1), pp. 9-32.
- [19] Elinson, A., Herrmann, J., Minis, I., Nau, D., Singh, G., (1997), “Toward Hybrid Variant/Generative Process Planning”, *Design Engineering Technical Conferences, DETC1997*, ASME, DFM-4333, Sacramento, CA.

- [20] Aha, D.W. (Ed.) (2003). *Mixed-initiative case-based reasoning: Papers from the ICCBR Workshop*. Trondheim, Norway: The Norwegian University of Science and Technology, Department of Computing Science.
- [21] Shimazu, H., Shibata, A., & Nihei, K. (1994). Case-based retrieval interface adapted to customer-initiated dialogues in help desk operations. *Proceedings of the 12th National Conference on Artificial Intelligence* (pp. 513-518) Seattle, WA: AAAI Press.
- [22] Branting, L.K. (1991). Building explanations from rules and structured cases. *International Journal of Man-Machine Studies*, **34**(6), 797-837.
- [23] Bergmann, R. (2002). *Experience management: Foundations, development methodology, and internet-based applications*. Berlin: Springer.
- [24] Lenz, M., & Ashley, K. (Eds.) (1998). *Textual case-based reasoning: Papers from the 1998 Workshop* (Technical Report WS-98-12). Menlo Park, CA: AAAI Press.
- [25] LUIKM (2003). [Language understanding for interactive knowledge management](#): A project at the Naval Research Laboratory.
- [26] Gupta, K., & Aha, D.W. (2003). Nominal Concept representation in sublanguage ontologies. *Proceedings of the 2nd International Workshop on Generative Approaches to the Lexicon* (pp. 53-62). Geneva, Switzerland: University of Geneva.
- [27] Aha, D.W. (1998). Feature weighting for lazy learning algorithms. In H. Liu & H. Motoda (Eds.) *Feature Extraction, Construction and Selection: A Data Mining Perspective*. Norwell, MA: Kluwer.
- [28] Branting, L., Hastings, J., & Lockwood, J. (1997). Integrating cases and models for prediction in biological systems, *AI Applications*, **11**(1), 29-48.
- [29] Hanney, K., & Keane, M. (1996). Learning adaptation rules from a case-base. *Proceedings of the 3rd European Workshop on Case-Based Reasoning* (pp. 179-192). Lausanne, Switzerland: Springer-Verlag.
- [30] Flemming, U, Coyne, R., & Snyder, J. (1994). Case-based design in the SEED system. *Proceedings of the 1st Congress on Computing in Civil Engineering* (pp. 446-453).
- [31] Leake, D., Maguitman, A., & Cañas, A. (2002). Assessing conceptual similarity to support concept mapping. *Proceedings of the 15th International Florida Artificial Intelligence Research Society Conference* (pp. 168-172). Menlo Park, CA: AAAI Press.
- [32] Avesani, P., Blanzieri, E., & Ricci, F. (1999). Advanced metrics for class-driven similarity search. *International Workshop on Similarity Search*. Florence, Italy.
- [33] Ullman, J. (1976). An algorithm for subgraph isomorphism, *Journal of the Association for Computing Machinery*, **23**(1), 31-42.

Author Biographies

JOSHUA D. SUMMERS is Assistant Professor of Mechanical Engineering at Clemson University. Dr. Summers has experience in the development of computer aided design systems and focuses research on collaboration and representation issues in engineering design. For more information see <http://www.vr.clemson.edu/credo/AID>.

BRUCE M. MCLAREN is a Systems Scientist at Carnegie Mellon University in Pittsburgh, PA. Dr. McLaren has an extensive background in both research and practical applications, specifically in knowledge-based systems, eCommerce, case-based reasoning, and artificial intelligence. For more information see <http://www.pitt.edu/~bmclaren>.

DAVID W. AHA leads NRL's Intelligent Decision Aids Group in Washington, DC. Dr. Aha's interests in AI include research and applications in case-based reasoning, mixed-initiative planning, and intelligent lessons learned processes/systems. For more information see <http://www.nrl.navy.mil/aic/iss/ida/>.