

Dynamically Structured Holographic Memory:  
A Hybrid of Discrete and Distributed Representation

By

Matthew F. Rutledge-Taylor

A thesis submitted to the Faculty of Graduate and Postdoctoral Affairs in partial  
fulfillment of the requirements for the degree of

Doctor of Philosophy

in

Cognitive Science

Carleton University

Ottawa, Ontario

© 2010, Matthew F. Rutledge-Taylor



Library and Archives  
Canada

Published Heritage  
Branch

395 Wellington Street  
Ottawa ON K1A 0N4  
Canada

Bibliothèque et  
Archives Canada

Direction du  
Patrimoine de l'édition

395, rue Wellington  
Ottawa ON K1A 0N4  
Canada

*Your file* *Votre référence*  
ISBN: 978-0-494-79637-5  
*Our file* *Notre référence*  
ISBN: 978-0-494-79637-5

**NOTICE:**

The author has granted a non-exclusive license allowing Library and Archives Canada to reproduce, publish, archive, preserve, conserve, communicate to the public by telecommunication or on the Internet, loan, distribute and sell theses worldwide, for commercial or non-commercial purposes, in microform, paper, electronic and/or any other formats.

The author retains copyright ownership and moral rights in this thesis. Neither the thesis nor substantial extracts from it may be printed or otherwise reproduced without the author's permission.

---

In compliance with the Canadian Privacy Act some supporting forms may have been removed from this thesis.

While these forms may be included in the document page count, their removal does not represent any loss of content from the thesis.

**AVIS:**

L'auteur a accordé une licence non exclusive permettant à la Bibliothèque et Archives Canada de reproduire, publier, archiver, sauvegarder, conserver, transmettre au public par télécommunication ou par l'Internet, prêter, distribuer et vendre des thèses partout dans le monde, à des fins commerciales ou autres, sur support microforme, papier, électronique et/ou autres formats.

L'auteur conserve la propriété du droit d'auteur et des droits moraux qui protègent cette thèse. Ni la thèse ni des extraits substantiels de celle-ci ne doivent être imprimés ou autrement reproduits sans son autorisation.

---

Conformément à la loi canadienne sur la protection de la vie privée, quelques formulaires secondaires ont été enlevés de cette thèse.

Bien que ces formulaires aient inclus dans la pagination, il n'y aura aucun contenu manquant.

  
**Canada**

## ABSTRACT

The purpose of this dissertation is to present Dynamically Structured Holographic Memory (DSHM), a system for modeling human memory. The dissertation consists of two parts. One component is theoretical in nature, while the other is practical. The aim of the theoretical component is to establish that a cognitive modeling niche for DSHM exists (Rutledge-Taylor, 2005; Rutledge-Taylor & West, 2007; West, Pyke, Rutledge-Taylor & Lang, 2010). The aim of the practical component is to prove that DSHM is a valid model of memory by demonstrating the ability to accurately model a variety of cognitive phenomena (Rutledge-Taylor, Pyke, & West, 2010; Rutledge-Taylor, Pyke, West & Lang, 2010; Rutledge-Taylor, Vellino, & West, 2008; 2010; Rutledge-Taylor & West, 2008). These aims are met by presenting evidence contained in a portfolio of previously published conference papers, technical reports, and new papers.

## ACKNOWLEDGMENTS

The work presented in this dissertation was made possible due to financial support in the form of the Derbyshire OGSST Award, Carleton University institute of cognitive science graduate scholarships, and a Carleton University doctoral bursary.

I have also received a tremendous amount of support in various forms from friends and family. In particular, special acknowledgement needs to go to my mother, Catherine Taylor, and my wife, Dawn Summer Langerak. Without their love and support this dissertation would never have been completed.

I dedicate this dissertation to my daughter, Michaela.

## TABLE OF CONTENTS

Abstract.....	ii
Acknowledgments.....	iii
List of Tables .....	x
List of Figures .....	xii
Dynamically Structured Holographic Memory: An Introduction to the Dissertation .....	1
Introduction.....	1
Main Portfolio Papers .....	8
Appendix Papers .....	23
Conclusions.....	26
References.....	28
Dynamically Structured Holographic Memory: The Mechanisms.....	42
Abstract.....	42
Introduction.....	42
Beagle .....	44
DSHM (Dynamically Structured Holographic Memory System) .....	52
Appendix.....	73
References.....	83
Dynamically Structured Holographic Memory: Cognitive Interpretation.....	85
Abstract.....	85
Introduction.....	85
Neural Plausibility .....	113
Appendix.....	115

References .....	118
Can ACT-R Realize “Newell’s Dream”? (Rutledge-Taylor, 2005) .....	122
Contribution .....	122
Abstract .....	122
Introduction .....	123
ACT-R .....	126
A Different Kind of Cognitive Architecture .....	136
ACT-R 6 .....	139
References .....	140
MALTA: Enhancing ACT-R with a Holographic Persistent Knowledge Store (Rutledge- Taylor & West, 2007) .....	143
Contribution .....	143
Abstract .....	143
Introduction .....	144
Holographic Associative Memory .....	146
MALTA .....	154
References .....	160
Modeling The Fan Effect Using Dynamically Structured Holographic Memory (Rutledge-Taylor & West, 2008) .....	164
Contribution .....	164
Abstract .....	164
Introduction .....	165
The Fan Effect .....	165

DSHM .....	167
DSHM Model of the Fan Effect.....	174
Evaluation of the Model.....	180
Relationship to the ACT-R Model.....	181
Discussion .....	184
References .....	185
Modeling a Three Term Fan Effect (Rutledge-Taylor, Pyke, West & Lang, 2010).....	187
Contribution .....	187
Abstract .....	187
Introduction.....	187
The Three Term Fan Experiment.....	188
Results.....	191
DSHM .....	195
The Model.....	196
In Summary.....	201
On-Going Work: Effect of How Fan is Distributed?.....	202
Appendix.....	204
References.....	206
A Holographic Model of Frequency and Interference: Rethinking the Problem Size Effect (Rutledge-Taylor, Pyke & West, 2010) .....	208
Contribution .....	208
Abstract .....	208
Introduction.....	209

Zbrodoff's Experiments.....	211
Re-Analysis of Zbrodoff's Experiments .....	214
Conclusions.....	222
References.....	222
Dynamically Structured Holographic Memory for Recommendation (Rutledge-Taylor, Vellino & West, 2010).....	224
Contribution.....	224
Abstract.....	224
Introduction.....	225
Experiments.....	237
Conclusion.....	245
Future Work.....	246
References.....	250
Using DSHM to Model Paper, Rock, Scissors .....	253
Contribution.....	253
Abstract.....	253
Dynamically Structured Holographic Memory .....	253
Existing Models of PRS.....	254
DSHM PRS Models.....	262
Conclusions.....	270
References.....	270
Appendix.....	272

Interference and ACT-R: New Evidence from the Fan Effect (West, Pyke, Rutledge-Taylor & Lang, 2010) .....	273
Contribution .....	273
Abstract .....	273
Introduction.....	273
Experimental Design.....	275
Method .....	276
Results.....	278
Conclusions.....	287
References.....	288
Cognitive Modeling Versus Game Theory: Why Cognition Matters (Rutledge-Taylor & West, 2004).....	289
Contribution .....	289
Abstract .....	289
Introduction.....	290
An Alternative Player Model .....	293
Rock=2.....	295
Modeling the Human results.....	301
Conclusions.....	304
References.....	305
ACT-R versus Neural Networks in Rock=2 Paper, Rock, Scissors (Rutledge-Taylor & West, 2005).....	308
Contribution .....	308

Abstract .....	308
Introduction: The Neural Network Models.....	309
ACT-R Models.....	313
The Results.....	318
Conclusions.....	319
References.....	320

## LIST OF TABLES

<i>Table 1.</i> Memory phenomena accounted for by DSHM.....	4
<i>Table 2.</i> Decay parameter comparison.....	56
<i>Table 3.</i> Default and recommended global parameters .....	69
<i>Table 4.</i> The number of associations that are computed when an unordered complex item is presented to a DSHM model. ....	71
<i>Table 5.</i> Default and recommended completions parameter values .....	72
<i>Table 6.</i> Completions of ?x.....	79
<i>Table 7.</i> Cosines of pBe (B.m) and the environmental vectors of the completions .....	80
<i>Table 8.</i> Relative values of the completions of ?x to the context item B .....	81
<i>Table 9.</i> Items and values for Completions of [B ?x].....	81
<i>Table 10.</i> The relative values of the completions of [B ?x] relative to the context item A .....	82
<i>Table 11.</i> Where are the hippies?.....	172
<i>Table 12.</i> Model reaction times and error rates for trues.....	180
<i>Table 13.</i> Model reaction times and error rates for falses.....	181
<i>Table 14.</i> Recall accuracy .....	192
<i>Table 15.</i> Recognition accuracy (%).....	192
<i>Table 16.</i> Recognition reaction time (ms/char).....	193
<i>Table 17.</i> Pairwise comparisons for correct reaction times .....	193
<i>Table 18.</i> Model recall accuracy .....	198
<i>Table 19.</i> Model reaction time (ms/char).....	200
<i>Table 20.</i> Experiment 1 & 2 group of problems .....	213

<i>Table 21.</i> Experiment 3 & 4 group of problems .....	214
<i>Table 22.</i> Humans versus networks .....	257
<i>Table 23.</i> Parameter values tested.....	263
<i>Table 24.</i> Models of human rock=1 PRS.....	266
<i>Table 25.</i> The game results of the human versus the neural network models .....	299
<i>Table 26.</i> Regression analysis on the performance of human subjects against the three network models.....	299
<i>Table 27.</i> Human and GA neural network model performances compared .....	312
<i>Table 28.</i> Naive ACT-R models of rock=1 paper, rock, scissors play .....	315
<i>Table 29.</i> Naive ACT-R models of rock=2 paper, rock, scissors play .....	316
<i>Table 30.</i> Savvy ACT-R models of rock=2 paper, rock, scissors play .....	319

## LIST OF FIGURES

<i>Figure 1.</i> Schematic diagram MALTA.....	16
<i>Figure 2.</i> The function defining the circular convolution, $t$ , of two vectors, $c$ and $x$ . .....	48
<i>Figure 3.</i> The function defining the circular correlation, $y$ , of two vectors, $c$ and $t$ . .....	51
<i>Figure 4.</i> The formula for calculating the expected cosine correlation between the memory vector of a word A, and the environmental vector of a word B, where $x$ is the number of vectors that have been added to A.m since B.e was added to A.m and the decay parameter is set to false. ....	55
<i>Figure 5.</i> Combine equation. ....	68
<i>Figure 6.</i> Confidence values for the top 10 completions of [green:?x:garage]. ....	75
<i>Figure 7.</i> Differences in the natural logarithms of the completion values of [green:?x:garage]. ....	75
<i>Figure 8.</i> Confidence values for the top 10 completions of [black:mug:?x]. ....	76
<i>Figure 9.</i> Differences in the natural logarithms of the completion values of [black:mug:?x]. ....	76
<i>Figure 10.</i> An ACT-R chunk representing the fact that $3 + 4 = 7$ . ....	92
<i>Figure 11.</i> An ACT-R production for remembering an addition fact. ....	94
<i>Figure 12.</i> An ACT-R production for writing an addition fact sum. ....	95
<i>Figure 13.</i> A DSHM item representing the fact that $3 + 4 = 7$ . ....	96
<i>Figure 14.</i> An incomplete item used for prompting the recall of an addition fact with one addend specified. ....	97
<i>Figure 15.</i> A representation of “the red car”, where words are the atoms of cognition. ...	98

<i>Figure 16.</i> A representation of “the red car”, where phonemes are the atoms of cognition. .....	98
<i>Figure 17.</i> A representation of “the red car”, where phonetic features are the atoms of cognition. ....	99
<i>Figure 18.</i> A formula for calculating the base level activation ( $B_i$ ) of a chunk $C_i$ , in ACT-R, where $n$ is the number of presentations of $C_i$ , $t_j$ is the time since the $j$ th presentation of $C_i$ , and $d$ is the decay parameter, which is usually set to 0.5.....	106
<i>Figure 19.</i> The natural logarithm of the canonical cosine formula in DSHM.....	106
<i>Figure 20.</i> Spreading activation, in ACT-R.....	108
<i>Figure 21.</i> A DSHM item representing the fact that $3 + 5 = 8$ .....	109
<i>Figure 22.</i> A DSHM item representing the incorrect proposition that $3 + 4 = 8$ . ....	110
<i>Figure 23.</i> The association strengths between two items as additional interfering associations are made. The $x$ axis represents the number of interfering associations. The $y$ axis represents association strength. One line depicts the trend when the decay parameter is set to true; the other when it is set to false. .	112
<i>Figure 24.</i> Combined associations to [borders:new_york] and [speaks:french]; darker shade indicates more association. ....	118
<i>Figure 25.</i> Reaction time for study set sentences. ....	178
<i>Figure 26.</i> Error rates for humans and simulations. ....	179
<i>Figure 27.</i> Reactions times for study set sentences. Bars indicate standard deviation..	182
<i>Figure 28.</i> Reaction times for foils. Bars indicate standard deviation.....	183
<i>Figure 29.</i> Recognition reaction time by sentence fan (ms/char) with confidence intervals.....	194

<i>Figure 30.</i> Recall accuracy (out of 16). .....	199
<i>Figure 31.</i> Recognition reaction times.....	201
<i>Figure 32.</i> Reaction times (ms/char) using the retrieval mechanism. ....	205
<i>Figure 33.</i> An example DSHM item representing an addition fact.....	216
<i>Figure 34.</i> An example DSHM item used for recalling the truth of an addition fact. ....	217
<i>Figure 35.</i> An example DSHM item used for recalling the correct sum of an addition fact. .....	217
<i>Figure 36.</i> True reaction times for human participants by addend in experiment 4.....	219
<i>Figure 37.</i> True reaction times for the model in experiment 4.....	219
<i>Figure 38.</i> True reaction times by addend for human participants in experiment 3.....	221
<i>Figure 39.</i> True reaction times by addend for the model in experiment 3. ....	221
<i>Figure 40.</i> Learning associations in DSHM. ....	232
<i>Figure 41.</i> Item Similarity in DSHM.....	232
<i>Figure 42.</i> DSHM state space.....	236
<i>Figure 43.</i> Decoding versus clustering in DSHM. ....	237
<i>Figure 44.</i> Top-N results for CF and DSHM on medicine collection. ....	244
<i>Figure 45.</i> Top-N results for CF and DSHM on biology collection.....	245
<i>Figure 46.</i> Perceptron networks.....	255
<i>Figure 47.</i> An example ACT-R chunk. ....	260
<i>Figure 48.</i> Points difference comparison for Rock=2 PRS. Human values include estimated 95% confidence values. ....	269
<i>Figure 49.</i> Strategy indices comparison for Rock=2 PRS.....	269

<i>Figure 50.</i> Reaction time in msec/character for responding false to a false cue as a function of the fan of the false item in the cue. ....	279
<i>Figure 51.</i> Percent errors for responding false to a false cue as a function of the fan of the false item in the cue. ....	280
<i>Figure 52.</i> The original Anderson & Reder (1999) ACT-R. ....	281
<i>Figure 53.</i> Re-plotted data from Anderson and Reder (1999). ....	282
<i>Figure 54.</i> A re-creation of the ACT-R fan model from. ....	283
<i>Figure 55.</i> The ACT-R (f) model fit to our data (note, that the I parameter could be increased to overlap the functions. ....	284
<i>Figure 56.</i> The ACT-R (f) model applied to the data from Anderson and Reder (1999). ....	286
<i>Figure 57.</i> The Anderson and Reder (1999) ACT-R fan model fit to our data for correctly identifying true cues only. ....	287
<i>Figure 58.</i> A comparison between the game theory solution and observed human play probabilities. ....	300
<i>Figure 59.</i> Points differences versus the three network opponents for both human subjects and the model. ....	303
<i>Figure 60.</i> Strategy indices versus the three network opponents for both human subjects and the model. ....	303
<i>Figure 61.</i> Human and model points differences versus the three network opponents. .	319

# DYNAMICALLY STRUCTURED HOLOGRAPHIC MEMORY: AN INTRODUCTION TO THE DISSERTATION

## Introduction

Cognitive Science as a discipline provides explanations for why cognitive phenomena occur and how they occur. The explanation for how a phenomenon occurs often involves a description of what processes underlie it. This need for process level accounts makes modeling a particularly useful tool in generating explanations for cognitive phenomena. Modeling takes many different forms and is applied in a variety of disciplines.

In this dissertation I will differentiate between these two categories of modeling systems. The term 'computational modeling' will be used to refer to modeling systems that are used, primarily, to model the computational level (Marr, 1982; Dawson, 1998), or knowledge (rational) level (Newell, 1990) of cognition. The term 'neurally inspired' will be used for systems that are used, primarily, to model the implementation level (Marr, 1982; Dawson, 1998) or biological level (Newell, 1990) of cognition. Most types of cognitive modeling systems that exist can be divided in to one of these two broad categories. The set of neurally inspired modeling systems include the simple perceptron (Rosenblatt, 1958), multilayer feed-forward networks using backpropagation learning (Rumelhart, Hinton & Williams, 1986), and self-organising maps (Kohonen, 1995). Examples of computational modeling systems are the cognitive architectures ACT-R (Anderson & Lebiere, 1998; Bothell, n.d.; Stewart & West, 2007), SOAR (Newell, 1990), and EPIC (Kieras & Meyer, 1997). The systems in each category have various strengths

and weaknesses, relative to one another, but ultimately complement each other well in providing a wide range of accounts for many of the phenomena of interest to cognitive scientists.

This distinction between these categories is driven mainly by the typical difference between these two groups in using distributed versus discrete representation. In neurally inspired modeling systems, information is typically represented as activation levels distributed across many units, such as nodes. In computational modeling systems, as I define them, information is typically localized in discrete units such as chunks.

Neurally inspired systems have a reputation for providing excellent models of perception and sensory processes, but have more difficulty modeling complex, rational behaviour (Boden, 2008; Dawson, 1998). In contrast, computational systems are typically used to produce models of complex rational behaviour, but usually provide only abstract accounts of sensation and the mechanisms underlying perception (Boden, 2008; Dawson, 1998). This indicates that using discrete units of information is best suited for representing conceptual knowledge, whereas using distributed units of information is best suited for modeling the transduction of information that exists in a continuous space.

This thesis examines this issue as it applies to memory. Specifically, one concern is that computational modeling systems are typically used only for generating one-off models of specific tasks, which don't require the human ability to retrieve knowledge learned in one context and apply it in a different context (Rutledge-Taylor, 2005). Another concern is that computational modeling systems do not seem to have the capability, in principle, to accommodate very large knowledge bases, something that even very young humans must be able to do (Rutledge-Taylor, 2005). Without addressing

these issues one cannot claim to have a fully functional model of the human cognitive architecture (Newell, 1990).

The aim of this thesis is to present a new system for modeling human memory that helps address these issues. The system is called Dynamically Structured Holographic Memory (DSHM). DSHM bridges computational and neurally inspired systems by representing concepts as discrete units with compositional structure, while representing the relationships between concepts in a distributed manner.

The ACT-R model of declarative memory (henceforth, DM) is used as a benchmark for evaluating DSHM. DM is one of the leading comprehensive models of declarative memory and it is integrated into a larger, well tested cognitive architecture (Anderson & Lebiere, 1998; Bothell, n.d.). The goal of creating DSHM is not to produce a memory system superior to DM, although advantages of DSHM are discussed. Rather it is to show that the basic functionality of DM, which is modeled based on empirical results and rational analysis (Anderson & Lebiere, 1998), can arise from a system which relies, internally, on distributed representation.

DSHM on its own does not solve the knowledge problems discussed above. For one, DSHM is not a full cognitive architecture. However, the case will be made that it has a role to play in solving them. It will be argued that an important distinction between task specific and general knowledge ought to be made in the cognitive modeling community, and that DSHM can accommodate this distinction. It will also be shown that DSHM is capable of reproducing several basic memory effects (Rutledge-Taylor, Pyke & West, 2010; Rutledge-Taylor, Pyke, West & Lang, 2010; Rutledge-Taylor & West, 2008), thus demonstrating that DSHM constitutes, at a minimum, a valid model of

memory, with the expressive power of ACT-R DM (see table 1). In addition, DSHM is shown to be able to handle very large knowledge bases, something that is required to model how humans deal with the vast store of information stored in memory (Rutledge-Taylor, Vellino & West, 2008; 2010).

Importantly, this thesis does not attempt to prove, by way of providing tested models, that DSHM can consume task specific knowledge, represent it internally in a context free manner, and then reproduce it in a format relevant to a given context. However, the claim that DSHM has the potential to do this is presented as a falsifiable hypothesis that will be tested in the future.

*Table 1.* Memory phenomena accounted for by DSHM

Memory Phenomena	Source
	Rutledge-Taylor & West, 2008; Rutledge-Taylor, Pyke, West
Fan effect	& Lang, 2010
Problem size effect	Rutledge-Taylor, Pyke & West, 2010
Large knowledge bases	Rutledge-Taylor, Vellino & West, 2008; 2010
Strategic game playing	Using DSHM to Model Paper, Rock, Scissors

*Format*

This dissertation consists of a portfolio of previously reviewed and published conference papers augmented with some additional, unpublished, papers. These papers fall in to four categories: 1) papers that support the theoretical claim that the

computational modeling systems that currently exist have inherent problems related to modeling how humans use their vast store of knowledge in flexible and context sensitive ways, and showing that DSHM has the potential to avoid these problems; 2) papers that demonstrate applications of DSHM to well known memory phenomena; 3) papers such as the current one that provide additional material that supports the papers in categories 1 and 2, in order to make this dissertation more cohesive; and, 4) additional papers included in the appendix. These additional papers should not be considered core to the dissertation, but are provided due to their relevance to the content of the papers in sections 1, 2, and 3. Included are two papers written by the author that are not otherwise widely available, and a paper relevant to the discussion of the performance of ACT-R in accounting for the fan effect.

For each paper in the portfolio that, a section of this introductory paper is devoted to: 1) summarizing the paper; and, 2) providing additional commentary to the paper. This commentary can serve several purposes, such as connecting related points made in different papers, or, highlighting how the paper contributes to the relevant theoretical or practical components of the dissertation.

Where papers in this portfolio have been published elsewhere, citation information is provided. Where no citation information is provided, the relevant paper is authored by Matthew Rutledge-Taylor, previously unpublished, and presented for the first time in this portfolio. The versions of the papers included in this portfolio may differ slightly from the originally published versions. These differences are limited to the correction of grammatical errors, spelling errors and other typographical errors etc, and to reformatting the papers to the format of this dissertation. The list of references provided

at the end of this paper is a compilation of all the references from all the papers in this portfolio.

### *Nomenclature Notes*

Throughout this dissertation some of the language used to describe technical events and processes is simplified so as to make the material herein more readable. For example, as described in *Dynamically Structured Holographic Memory: The Mechanisms*, a distinction is made between a *Completion* and a *completion*, where the former is a specific kind of data object, and the latter refers to the most salient component of the this object. The latter term is used primarily as a matter of convenience.

### *DSHM Versus a DSHM Model*

Only DSHM models, which are instances of the DSHM memory system, perform memory storage and retrieval. However, for brevity, the term DSHM is used when referring to properties of all DSHM models. For example, the phrase “the resonance value of a complex item in DSHM” is in reference to resonance in general, and not the particular resonance value of some particular complex item in some particular model. Thus this phrase is used instead of the more cumbersome, and technically, less precise, “the resonance values of complex items in DSHM models”, which refers to the set of possible resonance values.

### *Associations*

Another significant convenience is with respect to references to items’ associations with one another. As described in many of the papers in this portfolio, complex DSHM items can contain either ordered lists of sub-items, or unordered sets of

sub-items. Whether the items bear an ordered or unordered relationship to one another fundamentally affects the nature of the associations between the sub-items. Thus, to say that two items are associated together does not provide any information about the nature of the association. However, it is not convenient to be precise about the particular associations between items in most contexts.

Therefore, the reader is to understand that when the fact that two items are associated together is mentioned in any paper in this portfolio, the meaning is that the items bear either an ordered or unordered relationship to one another and that which kind of relationship they bear to one another has not been specified and the details thereof have been omitted.

#### *Presenting an Item to DSHM*

The language used for storing to, and retrieving knowledge from memory is somewhat inconsistent in this dissertation. This stems from the fact that DSHM is not a complete cognitive architecture, and does not include any mechanisms for perception and action (e.g., reporting retrieved memories). As such, it would be incorrect to refer to a DSHM model as perceiving objects, reading sentences, studying facts, or otherwise entertaining concepts. Thus, the preferred term for inputting information into a DSHM model is 'present'. To say that an item is presented to a DSHM model is a neutral way of stating that the DSHM model has ingested the item, and that it has been learned or reinforced.

Since memory retrieval is based, primarily, on completing incomplete items, the preferred way of referencing this process is to say that an incomplete item is presented to

a DSHM model, or that a model is caused to complete an incomplete item. This is understood as a neutral way of referring to the DSHM mechanisms for queued recall.

Despite this defence of preferred neutral terms, the imprecise (and perhaps, technically incorrect) use of the terms perceive, read, and study appear in the papers included in this dissertation portfolio. In these cases, the models described should not be understood as possessing any perceptual capabilities beyond the core mechanisms described in the official account of DSHM contained herein.

### Main Portfolio Papers

#### *Dynamically Structured Holographic Memory: The Mechanisms*

##### *Summary*

The mechanics of DSHM are presented in this paper. Items are the basic units of information. They can be complex (e.g., [fast red car]) or atomic (e.g., 'red'); complete (e.g., [fast red car]) or incomplete (e.g., [fast red ?x]). Complex items are items that contain a set of other items. The set of sub-items of a complex item is either ordered (e.g., [1:2:3:4]) or unordered (e.g., [red green blue]). Incomplete items contain, as a sub-item, either another incomplete item or a query item (e.g., '?x'), which is an unspecified item that acts like a variable.

An instance of DSHM (e.g., a model) is created by supplying the DSHM constructor with four parameters, the principle of which is *vector length*, which determines how many elements are in the vectors representing items in the DSHM model. Vector length determines the memory capacity of the model.

Information is entered into DSHM by presenting an instance of DSHM with items. When an item is presented to DSHM the sub-items of the item are associated together. Information is retrieved by presenting an instance of DSHM with an incomplete item. Information about what items could be substituted for the query items of the incomplete item is extracted from the complete sub-items of the item (aka context items). The DSHM model then generates a rank ordered list of completions of the incomplete item. Each completion is paired with a completion value which indicates how strong the completion is. The list of retrieved completions can be limited to a cluster of the best completions. This is used to allowing a DSHM model to return more than one answer (but not an exhaustive list) to a recall question.

Instead of making use of a concept such as base-level activation, as is found in ACT-R, a DSHM model can compute the resonance of a complex item. This resonance value is a measure of the strengths of the associations between the sub-items of the item. An item that has been reinforced will score a high resonance, while items that have not will score low resonance values.

### *Commentary*

The mechanisms that underlie DSHM are very different from those of ACT-R DM. However, functionally, DSHM and DM are very similar.

Both systems are able to assign strength values to complex units of information; this is the resonance value of a complex item in DSHM, and the activation of a chunk in ACT-R.

Both systems provide mechanisms for satisfying variables based on supplied values in complex units of information. In DSHM this is accomplished by completing an

incomplete item. In ACT-R this occurs in both the left-hand-side and right-hand-side of production rules.

The similarities and differences between DSHM and DM, as they pertain to interpreting cognitive phenomena are discussed in *Dynamically Structured Holographic Memory: Cognitive Interpretation*.

*Dynamically Structured Holographic Memory: Cognitive Interpretation*

*Summary*

The primary purpose of this paper is to provide a relatively high level discussion of the theory behind DSHM. DSHM is described as extending the basic principles of BEAGLE in order to make it a general purpose memory system. The basic features of DSHM, such as the ad hoc nature of items, and learning, are described.

Various comparisons between ACT-R and DSHM are made, including differences in how each system: defines the atoms of cognition; computes and interprets activation; and, accommodates reinforcement, noise, and, memory degradation.

*Commentary*

As is the case with the mechanisms that underlie DSHM and ACT-R, the cognitive interpretation of DSHM and ACT-R have some differences as well as some similarities. A general theme that is worth taking note of is the use of variables in ACT-R to account for psychological effects that arise as emergent properties in DSHM.

For example, noise exists in both systems. In ACT-R it is governed by explicit variables. In DSHM it is an emergent property of overloading the memory vectors of items.

Decay exists in both systems. Again, in ACT-R decay is parameterized by an explicitly defined variable. In DSHM, the decay of old knowledge is the result of interference with new knowledge.

However, the greatest difference between ACT-R and DSHM is with respect to the use of chunk types and the persistence of chunks. It is the ad hoc nature of items in DSHM that gives it the potential for greater flexibility than ACT-R DM. In the very least, the fact that DSHM operates using very different mechanisms than does ACT-R, gives it the ability to provide different explanations for phenomena captured by both systems.

### *Can ACT-R Realize “Newell’s Dream”?*

#### *Citation Information*

Rutledge-Taylor, M. F. (2005). Can ACT-R realize “Newell’s Dream”? In B. G. Bara, L. Barsalou & M. Bucciarelli (Eds.) *Proceedings of the 27<sup>th</sup> Annual Conference of the Cognitive Science Society* (pp.1895-1900). Stresa, Italy: University and Polytechnic of Turin.

#### *Summary*

In this paper my weak and strong criteria for realizing Newell’s dream of a unified theory of cognition are defined. The weak criterion is that *a unified theory of cognition should provide a common set of tools and principles such that for any psychological phenomenon a model that accounts for the phenomenon can be created.* The strong criterion is that *a unified theory of cognition should provide a common set of tools and principles such that a single model that accounts for every psychological*

*phenomenon can be created.* The strong criterion asserts that a theory of cognition is not truly unified until the ability to retrieve knowledge and apply it in any arbitrary context is understood. It is argued that the goal of cognitive modeling ought to be to meet the strong criterion.

It is argued that, while ACT-R satisfies the weak criterion, it does not satisfy the strong criterion, due to two limitations of its declarative memory system. The first is that chunks are limited in terms of how much information they can store. Unofficially, each chunk should contain only a few slots up to a maximum of seven, plus or minus two (Miller, 1956). Thus, a single chunk cannot store all the relevant knowledge someone has of a thing. Storing knowledge of a thing in relational chunks (such as “john loves mary”) is not an adequate solution because of the second limitation. This is that updating DM in order to assimilate new knowledge could easily entail discarding and creating too many new chunks than is feasible in ACT-R.

It is suggested that a hybrid architecture based on ACT-R and Barsalou’s situated simulation theory (Barsalou, 1999; Barsalou, 2003; Barsalou, Simmons, Barbey, Wilson, 2003), would be able to meet the strong criterion.

#### *Commentary*

From a theoretical perspective, the reason ACT-R does not satisfy the strong criterion is because it does not make a distinction between an atom of thought and an atom of knowledge. A side-effect of this ontological commitment is that DM is not suitable for: consuming context specific knowledge; transforming it into general purpose context-free information and storing it; and, then being able to retrieve context-free knowledge and make it relevant to a different context. This claim pertains to the details

of information manipulation and can be interpreted as supporting Dreyfus' arguments presented in the paper, from a mechanical perspective.

ACT-R chunks exist only for encoding information relevant to a particular task, and are therefore wedded to a particular context. From a mechanical perspective, the issue is with the fact that an ACT-R model must specify a finite set of chunk types that constrain the structures of information that can be stored in DM. In contrast, all complex items in DSHM are ad hoc and are not constrained by a finite set of item types. This provides DSHM with the flexibility required to apply knowledge acquired in one context to a different context. See the *Geography Fact Conversion* example in *Dynamically Structured Holographic Memory: Cognitive Interpretation*. DSHM is able to achieve this flexibility because the information required to retrieve items from memory is stored in a distributed, not discrete manner.

It should be noted that this paper was written before DSHM had been created. Therefore, the arguments for the limitations of ACT-R do not include any references to the relative strengths of DSHM. However, an aim of this dissertation is to make the case the DSHM has the potential to satisfy this criterion.

*MALTA: Enhancing ACT-R with a Holographic Persistent Knowledge Store*

*Citation Information*

Rutledge-Taylor, M. F. & West, R. L. (2007). MALTA: Enhancing ACT-R with a holographic persistent knowledge store. In D. S. McNamara & J. G. Trafton (Eds.), *Proceedings of the 29<sup>th</sup> Annual Conference of the Cognitive Science Society* (pp. 1433-1438). Nashville, TN: Cognitive Science Society.

*Summary*

An early version of DSHM is introduced. The possibility of augmenting ACT-R with DSHM to create a hybrid architecture called Multiple and Long-term Task Architecture (MALTA) is presented.

Two variations on MALTA are discussed. The common features are the use of DSHM as a long-term memory store and ACT-R as the architecture for performing tasks. Each MALTA model consists in a set of ACT-R model templates which define the chunk types and productions used by the model. Given a specific context, MALTA invokes an instance of a relevant ACT-R model template, and prompts DSHM to furnish the model with a set of task specific chunks which conform to the chunk types defined by the model; see figure 1.

According to this paradigm, a DSHM model of long-term memory persists for the duration of the MALTA model. In contrast, the ACT-R models invoked for performing tasks are temporary. MALTA learns by exporting the contents of DM to DSHM after each ACT-R model halts.

*Commentary*

MALTA exists only as a paper model in that it has yet to be formalized in code. However, despite the fact that DSHM has evolved somewhat since MALTA was proposed, the framework of how MALTA would work is still applicable. Writing a computer program which actualizes MALTA constitutes a medium term goal in the future development of DSHM.

Whether it be MALTA, or some other framework, it is firmly my belief that a fruitful marriage between DSHM and ACT-R (or, some other computational cognitive

architecture) is possible, and will lead to more robust models of learning and intelligent action. Such a merger would marry the established proven theory of ACT-R, including its production system(s), with the flexible declarative knowledge representation of DSHM. The product would be a system that might satisfy the strong criterion specified in Rutledge-Taylor (2005).

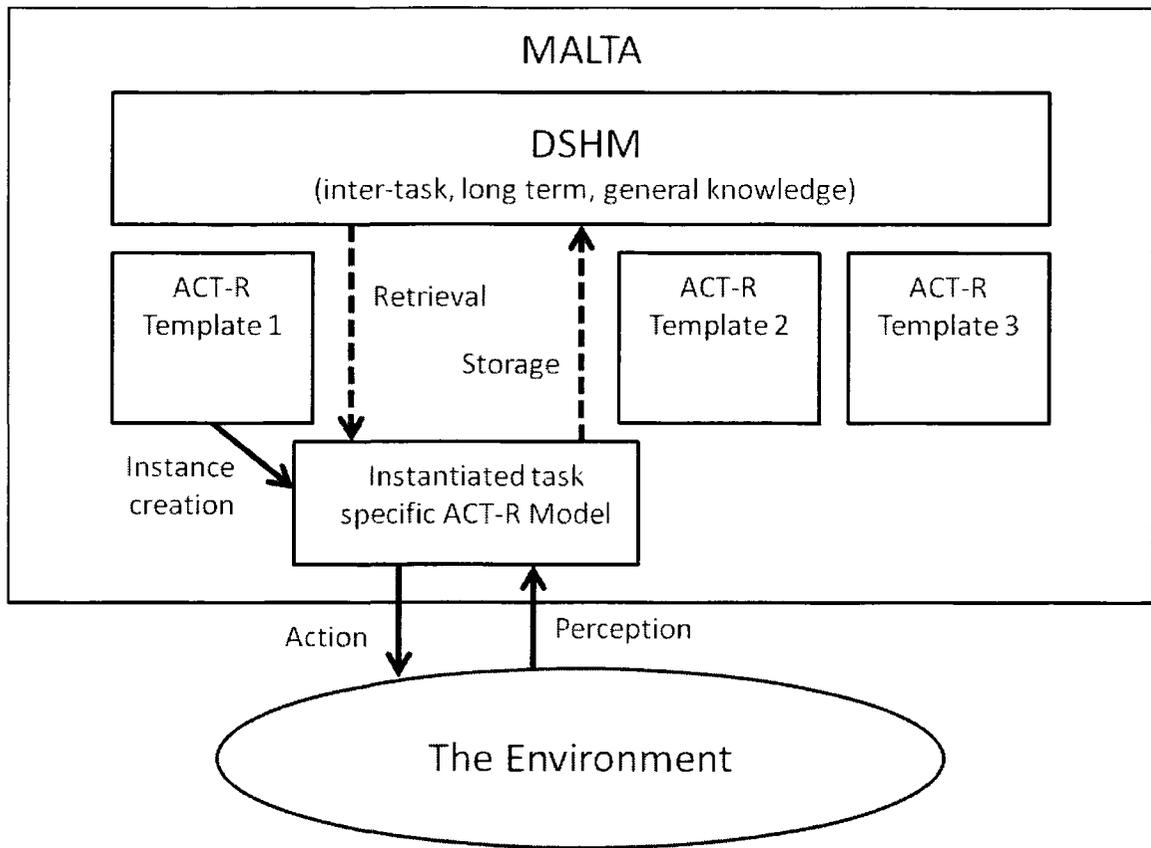


Figure 1. Schematic diagram MALTA

*Modeling The Fan Effect Using Dynamically Structured Holographic Memory*

*Citation Information*

Rutledge-Taylor, M. F. & West, R. L. (2008). Modeling The fan effect using dynamically structured holographic memory. In B. C. Love, K. McRae, & V. M. Sloutsky (Eds.), *Proceedings of the 30<sup>th</sup> Annual Conference of the Cognitive Science Society* (pp. 385-390). Washington, DC: Cognitive Science Society.

*Summary*

A DSHM model of Anderson's original fan effect experiment is discussed. The model is able to account for how participants provide multiple answers to recall questions

(e.g., “name all of the places where hippies can be found”). It also provides a good fit to the recognition reaction time data (i.e., the model can produce similar profile of RT values to those produced by human experimental participants).

Some of the mechanisms that underlie the model’s performance are compared to the corresponding mechanisms in ACT-R. For example, the confidence value of a completion is comparable to the activation of a recalled chunk. It is suggested that DSHM can provide process level accounts of some mathematically defined features of the declarative memory system of ACT-R.

#### *Commentary*

The importance of this paper is twofold. First, it presents the first real test of DSHM as a valid system for modeling memory. The second is that the comparison of DSHM and ACT-R revealed some differences in the theory upon which the systems are based.

For example, for DSHM and ACT-R, models of the fan effect generate reaction times based on the activations of the propositions (items in DSHM and chunks in ACT-R). However, for DSHM this value is based only on the degree of association between the terms in the proposition (i.e., the strength of the association between the person and the place and vice-versa). That is, the fan of a proposition is a matter of the relationship between the person and the place.

In ACT-R, the fan effect is based on spreading activation. Thus, the fan of a proposition is captured by the relationships between chunks in DM.

*Modeling a Three Term Fan Effect**Citation Information*

Rutledge-Taylor, M. F., Pyke, A. A., West, R. L. & Lang, H. (2010). Modeling a three term fan effect. In D. D. Salvucci & G. Gunzelmann (Eds.), *Proceedings of the Tenth International Conference on Cognitive Modeling* (pp. 211-216). Philadelphia, PA: Drexel University.

*Summary*

The results of a fan effect experiment are reported. Participants studied a set of sentences with three content terms. Each term appeared in either one or four sentences, resulting in sentence fans of three, six, nine and 12.

In a recall phase, incomplete sentences were completed with an accuracy that decreased with sentence fan. In a recognition phase, reaction time increased with sentence fan. The reaction time for correct rejections was higher than that for correctly affirming true sentences.

The DSHM model produced a good fit to the human data on both the recall and recognition tasks.

*Commentary*

Producing a good fit to the recognition task proved to be a challenge. The reaction time mechanism created in Rutledge-Taylor and West (2008), included in this portfolio, provided an excellent fit to the true correct reaction times. However, it predicted a very low RT value for the correct rejection of false sentences with a fan of 12; this was the highest average RT for human participants. The reason for this is discussed in the paper. As a result of this anomaly a new mechanism for determining the reaction

times for the model was developed. This produced a better fit for the correct rejections, but fit the correct affirmations less well.

It is my opinion that the more methodologically sound mechanism is the one created for Rutledge-Taylor and West (2008), and that the odd correct rejection predication is simply an artefact of the counterbalancing of the stimuli. However, artefact or not, this prediction needs to be explained, and an account of why the human participants did not produce the results predicted by the model needs to be provided. Improving upon the official mechanisms for producing quantifiable model output is a short-term future work goal.

*A Holographic Model of Frequency and Interference: Rethinking the Problem Size Effect*

*Citation Information*

Rutledge-Taylor, M. F., Pyke, A. A., & West, R. L. (2010). A holographic model of frequency and interference: Rethinking the problem size effect. Carleton University Cognitive Science Technical Report 2010-02. URL <http://www.carleton.ca/ics/TechReports>

*Summary*

A DSHM model of the problem size effect (Zbrodoff & Logan, 2005) is presented. The model is based on the experiments presented in Zbrodoff (1995), where the contributing effects of problem frequency and problem interference were tested. The DSHM model was found to be able to account for both problem frequency and interference. However, the DSHM model was not able to provide a satisfactory account

of all the human data. This was attributed to behavior on the part of the human participants that lay outside of simple memory recall (i.e., the use of strategies etc.)

### *Commentary*

This paper helps demonstrate that the fan effect is robust and can occur in areas of cognition outside of the classic paradigm of linguistic fact memorization. In this case, it is shown that interference effects contributing to the problem size effect can be explained as a special case of the fan effect.

This paper also demonstrates that DSHM can model frequency effects. That is, when a fact is studied more frequently than another fact, it is recalled more quickly than a fact that has been reinforced less frequently. Frequency in ACT-R is accounted for in the base-level activation of chunks.

### *Dynamically Structured Holographic Memory for Recommendation*

#### *Citation Information*

Rutledge-Taylor, M. F., Pyke, A. A., & West, R. L. (2010). A holographic model of frequency and interference: Rethinking the problem size effect. Carleton University Cognitive Science Technical Report 2010-02. URL <http://www.carleton.ca/ics/TechReports>

#### *Summary*

DSHM is used to create three models of recommendation: one for movies, and two for journal articles. In all three cases, a DSHM model was fed a very large corpus of data, and was prompted to make recommendations. The DSHM models were

competitive in accuracy for predicting movie preferences and more accurate than collaborative filtering on the very sparse journal article data sets.

### *Commentary*

This paper supports the claim that DSHM models can operate on large knowledge bases (tens of thousands of items). This ability is essential for building models that contain an amount of general knowledge equivalent to what a normal adult human has acquired in his or her lifetime.

Another important contribution that this paper makes is in describing ‘the cluster method’, which is the only example of memory vector comparison in this portfolio. Using the memory vector of one item to discover other items that share a similar semantic content is a powerful feature of DSHM. In this paper, it was used to generate better movie recommendations than were produced using the ‘decoding method’, which corresponds to ordinary memory recall in DSHM.

### *Using DSHM to Model Paper, Rock, Scissors*

#### *Summary*

DSHM is used to build models of playing a variation of the game Paper, Rock, Scissors (PRS), where winning with rock is worth twice as much as other wins. In order to play PRS to win, one must be able to detect sequential dependencies in one’s opponents’ play choices (West & Lebiere, 2001). Additionally, to win at the rock=2 version of PRS, players must attempt to maximize wins when playing rock, while minimizing losses when playing scissors (Rutledge-Taylor & West, 2004).

The DSHM model is able to detect sequential dependencies by storing items representing sequences of opponents' play. Making correct predictions of opponents' play requires that the model be able to adjust the internal association strengths between the sub-items representing the predicted play and the items representing the previous lags (i.e., plays), appropriately. The model is able to maximize rocks wins and minimize scissors losses by reinforcing some observed sequences more than others.

The DSHM models are compared to ACT-R and perceptron-like neural network models. All three kinds of models were able to fit the human data reasonably well. However, DSHM differed from the ACT-R models and the perceptron models in that they produced a good fit on one metric (strategy index versus the lag 2 network opponent) that the other models were not able to, but failed on another metric (points difference versus the lag 2 network opponent) that both the ACT-R and perceptron models were able to fit.

#### *Commentary*

There are two interesting conclusions that can be drawn from this paper. The first is that it demonstrates that DSHM is able to model a dynamic short-term memory task. DSHM was designed to model long-term memory, the contents of which would be relatively stable. In contrast, success at PRS requires that relative priority be given to more recent facts about opponents' play, and that inconsistent facts can be reconciled appropriately.

For example, it may have been the case earlier in the game that an opponent tended to follow the sequence of plays, *paper*, *paper* with *rock*, while later in the game following *paper*, *paper*, with *scissors*. The more relevant recent fact must be recalled more readily than the old fact, until it too becomes out of date.

The fact that DSHM was able to produce good models of PRS play was surprising, because DSHM lacks a mechanism for temporal decay. That is, old facts do not decay simply because they are old; they must be interfered with by new facts. This result suggests that perhaps common memory mechanisms are responsible for both short-term and long-term memory.

The other interesting conclusion drawn from this paper is that it provides an example of a task that both DSHM and a neural network are able to model. This helps support the claim that DSHM connects to neurally inspired systems. However, the fact that an ACT-R model of this task also exists, refutes any claim that models of this task support a tighter connection between DSHM and neurally inspired modelling systems, than between ACT-R and neurally inspired modelling systems.

## Appendix Papers

### *Interference and ACT-R: New Evidence from The Fan Effect*

#### *Citation Information*

West, R. L., Pyke, A. A., Rutledge-Taylor, M. F. & Lang, H. (2010). Interference and ACT-R: New evidence from the fan effect. In D. D. Salvucci & G. Gunzelmann (Eds.), *Proceedings of the Tenth International Conference on Cognitive Modeling* (pp. 277-281). Philadelphia, PA: Drexel University.

#### *Summary*

A version of Anderson's classic fan effect experiment (1974) was run using sentences with three content terms each (Rutledge-Taylor, Pyke, West & Lang, 2010).

The human data clearly showed an exponential upward curve in the recognition reaction times, when plotted against the fans of the test sentences.

Contrary to the ACT-R theory of spreading activation, the fans of all three terms in false sentences were proven to affect the reaction times for these sentences.

An ACT-R model of this experiment was able to fit the true correct RT data well, but a satisfactory account of the false correct data was elusive.

#### *Commentary*

An important finding presented in this paper is the fact that the fans of all the terms in a sentence affect recognition reaction time of the sentence. ACT-R does not predict this. However, DSHM does. A DSHM model of this experiment is discussed in Rutledge-Taylor, Pyke, West and Lang (2010), which is included in this portfolio.

This finding supports the priority DSHM places on interference as the main feature of memory responsible for accounting for many memory phenomena (See *Dynamically Structured Holographic Memory: Cognitive Interpretation*).

#### *Cognitive Modeling Versus Game Theory: Why Cognition Matters*

#### *Citation Information*

Rutledge-Taylor, M. F. & West, R. L. (2004). Cognitive modeling versus game theory: Why cognition matters. In M. Lovett, C. Schunn, C. Lebiere & P. Munro (Eds.), *Proceedings of the Sixth International Conference on Cognitive Modeling* (pp. 255-260). Pittsburgh, PA: Carnegie Mellon University/University of Pittsburgh.

*Summary*

The relationship between game theory predictions and a cognitive model of game playing based on the detection of sequential dependencies was investigated. A perceptron-like neural network model of a modified version of Paper, Rock, Scissors (PRS) was created. The performance of this model was compared to that of human participants. The model and the participants were pitted against three unique neural network players in a version of PRS where winning by playing rock was worth two points, while other wins were worth only one point each. It is demonstrated that game theory (VonNeumann & Morgenstern, 1944) cannot account for the behaviour of individual players in this version of PRS.

*ACT-R versus Neural Networks in Rock=2 Paper, Rock, Scissors*

*Citation Information*

Rutledge-Taylor, M. F. & West, R. L. (2005). ACT-R versus neural networks in rock=2 paper, rock, scissors. *Proceedings of the Twelfth Annual ACT-R Workshop*, 19-23. Trieste, Italy: Universita degli Studi di Trieste.

*Summary*

The work presented in Rutledge-Taylor & West (2004) is continued. An ACT-R model of a modified version of Paper, Rock, Scissors (PRS) is presented. The question of how to create an ACT-R model where rewards of various magnitudes need to be implemented is investigated. The model presented makes use of some simple techniques such as the double retrieval and harvesting of chunks, and the manipulation of the default

parameter for noise ( $\sigma$ ), to accurately account for human performance in this version of PRS.

### Conclusions

The papers in this portfolio converge in supporting the two main goals of this dissertation. The first was to establish there is a niche for DSHM in cognitive modeling. The second was to demonstrate that DSHM can be used to produce valid models of memory. The latter of the two goals was easier to establish. It has been demonstrated that DSHM is able to provide accounts of several memory phenomena: the fan effect (Rutledge-Taylor & West, 2008; Rutledge-Taylor, Pyke, West & Lang (2010); the problem-size effect (Rutledge-Taylor, Pyke & West, 2010); recommendation (Rutledge-Taylor, Vellino & West, 2010); and, game playing (Using DSHM to Model Paper, Rock, Scissors). No finite list of modeling successes can prove that a system is capable of modeling any cognitive phenomenon. Indeed, there will certainly be phenomena that DSHM has more difficulty in modeling than others. Nevertheless, this set of early successes suggests that DSHM captures some important aspects of human memory and is worthy of further study and use in generating cognitive models.

Making the case that there is a niche for DSHM draws evidence of two varieties. The first is the argument that there is a potential problem with existing cognitive architectures (Rutledge-Taylor, 2005; Schultheis, Barkowsky & Bertel, 2006), and that it needs to be addressed. This is a general problem with applying knowledge in contexts different from the ones in which it is acquired.

The second is the argument that DSHM possesses different capabilities than existing architectures (West, Pyke, Rutledge-Taylor & Lang 2010; Dynamically Structured Holographic Memory: Cognitive Interpretation), and may be able to solve the issue identified (Rutledge-Taylor & West, 2007). The use of distributed representations, internally, supports the ad hoc nature of items in DSHM. This lack of item types is the primary difference between DSHM and other chunk based architectures. It is also what gives DSHM the potential to solve the problem of knowledge context dependence.

#### *Additional Discussion*

The greatest mistake that a cognitive modeler can make is to reject all of the existing modeling theory and set out to ‘correct’ all previous mistakes by creating a new, perfect, modeling system. Doing so runs the risk of throwing the baby out with the bath water, so to speak. Additionally, by rejecting existing theory, any new theory, or modeling system, will have little to connect it to the discipline.

I have deliberately avoided this mistake by focusing on how existing modeling systems can be improved upon, and by explicitly connecting DSHM to existing systems. However, a potential criticism of DSHM is that it is too ambitious and that trying to connect neurally inspired system with computation systems to too complex a task.

McClelland espouses the virtues of simplification in modeling (McClelland, 2009). He argues that models ought to be easily understood so as to make interpreting the performance of the model transparent. He acknowledges that simplification includes the risk of eliminating essential properties that may turn out to have been relevant to understanding the target phenomenon.

I agree with McClelland's point, which is that determining how the structure of the model and the performance of the model relate to one another must be a tractable problem. However, this does not entail that a model must be simple for this to be the case. Rather, it is the case that for a model to be easily understood, it must vary from an available reference point by an incremental amount. If the reference point is only a basic understanding of the modeling system or the relevant cognitive phenomena, the model must be simple. However, where a series of models make incremental, small changes to one another, a very complex model can be understood by tracing the series of models to a point of common knowledge.

Thus DSHM, despite its complexity, is not guilty of violating McClelland's maxim. From a mechanical perspective, DSHM is similar to BEAGLE; and, from a psychological perspective, it is similar to ACT-R DM. DSHM, the system, although novel in various respects is comprehensible due to its similarity to existing systems. Additionally, specific DSHM models often admit of direct comparisons to existing models created using other modeling systems.

#### References

- Adomavicius, G., & Tuzhilin, A. (2005), Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *Knowledge and Data Engineering, IEEE Transactions on*, 17(6), 734–749.
- Anderson, J. R. (1974). Retrieval of propositional information from long-term memory. *Cognitive Psychology*, 6, 451-474.

- Anderson, J. R. (1977). Memory for information about individuals. *Memory and Cognition*, 5, 430-442.
- Anderson, J. R., & Lebiere, C. (Eds.) (1998). *The atomic components of thought*. Mahwah, NJ: Lawrence Erlbaum.
- Anderson, J. R. & Matessa, M. (1998). The rational analysis of categorization and the ACT-R architecture. In M. Oaksford & N. Chater (Eds.), *Rational models of cognition*. Oxford University Press.
- Anderson, J. R. & Reder, L. R. (1999). The fan effect: New results and new theories. *Journal of Experimental Psychology: General*, 128(2), 186-197.
- Atkinson, R. C., & Shiffrin, R. M. (1968). Human memory: A proposed system and its control processes. In K. W. Spence and J. T. Spence (Eds.), *The Psychology of learning and motivation: Vol 2. advances in research and theory*, (pp. 89-195). New York: Academic Press.
- Baddeley, A. D. & Hitch, G. (1974). Working memory. In G. H. Bower (ed.) *Recent advances in learning and motivation: Vol. 8.* (pp. 47-90). New York: Academic Press.
- Barsalou, L. W. (1983). Ad hoc categories. *Memory & Perception*, 11, 211-227.
- Barsalou, L. W. (1999). Perceptual symbol systems. *Behavioral and Brain Sciences*, 22, 577-609.
- Barsalou, L. W. (2003). Situated simulation in the human conceptual system. *Language and Cognitive Processes*, 18, 513-562.

- Barsalou, L. W., Simmons, W. K., Barbey, A. K., and Wilson, C. D. (2003). Grounding conceptual knowledge in modality-specific systems. *TRENDS in Cognitive Science*, 7, 84-91.
- Boden, M. A. (2008). An evaluation of computational modeling in cognitive science. In R. Sun (Ed.), *The cambridge handbook of computational psychology*. (pp. 667-683). New York: Cambridge University Press.
- Bothell, D. (n.d.). *ACT-R 6.0 reference manual*. Retrieved August 11, 2010, from <http://act-r.psy.cmu.edu/actr6/reference-manual.pdf>
- Bothell, D., Byrne, M. D., Lebiere, C., & Taatgen, N. A. (2004). *ACT-R 6 proposals*. Retrieved from <http://act-r.psy.cmu.edu/act-r6/ACT-R6proposal.pdf>
- Bunting, M. F., Conway A. R. A., & Heitz, R. P. (2004). Individual differences in the fan effect and working memory capacity. *Journal of Memory and Language*, 51(4), 604-622.
- Burke, R. (2002). Hybrid recommender systems: survey and experiments. *User Modeling and User-Adapted Interaction*, 12(4), 331–370.
- Byrne, M. D., & Anderson, J. R. (1998). Perception and action. In J. R. Anderson & C. Lebiere (Eds.), *The atomic components of thought*, (pp. 167-200). Mahwah, NJ: Lawrence Erlbaum.
- Churchland, P. S., & Sejnowski, T. J. (1992). *The computational brain*. Cambridge, MA: The MIT Press.
- Clark, A. (1997). *Being there: Putting brain, body, and world together again*. Cambridge, MA: The MIT Press.

- Craik, F. I. M. & Lockhart, R. S. (1972). Levels of processing: A framework for memory research. *Journal of Verbal Learning and Verbal Behavior*, *11*, 671-684.
- Dawson, M. R. W. (1998). *Understanding cognitive science*. Malden, MA: Blackwell.
- Dreyfus, H. L. (1979). From micro-worlds to knowledge representation: AI at an impasse. In J. Haugeland (Ed.), *Mind design II* (pp. 143-182). Cambridge, MA: The MIT Press.
- Eliasmith, C., & Anderson, C. H. (2003). *Neural engineering: Computation, representation and dynamics in neurobiological systems*. Cambridge, MA: MIT Press.
- Estes, W. K. (1972). Research and theory on the learning of probabilities. *Journal of the American Statistical Association*, *67*, 81-102.
- Frias-Martinez, E., Magoulas, G., Chen, S., & Macredie, R. (2006). Automated user modeling for personalized digital libraries. *International Journal of Information Management*, *26*(3), 234-248.
- Fudenburg, D., & Levine, D. K. (1998). *The theory of Learning in Games*. Cambridge, MA: The MIT Press.
- Gazzaniga, M. S. (1998). The split brain revisited. *Scientific American*, July, 50-55.
- Gentner, D., Holyoak, K. J., & Kokinov, N. (Eds.) (2001). *The analogical mind: Perspectives from cognitive science*. Cambridge, MA: MIT Press.
- Goetz, P. & Walters, D. (2000). A neuronal basis for the fan effect. *Cognitive Science*, *24*(1), 151-167.
- Hamann, M. & Ashcraft, M. (1986). Textbook presentations of the basic addition facts. *Cognition and Instruction*, *3*, 173-192.

- Herlocker, J. L., Konstan, J. A., Terveen, L. G., & Riedl, J. T. (2004). Evaluating collaborative filtering recommender systems. *ACM Transactions on Information Systems*, *22(1)*, 5–53.
- Huang, Z., Chen, H., & Zeng, D. (2004). Applying associative retrieval techniques to alleviate the sparsity problem in collaborative filtering. *ACM Transactions on Information Systems*, *22(1)*, 116–142.
- Huggett, M., Hoos, H., & Rensink, R. (2007). Cognitive principles for information management: The principles of mnemonic associative knowledge (p-mak). *Minds and Machines*, *17(4)*, 445–485.
- Jones, M. N., & Mewhort, D. J. K. (2007). Representing word meaning and Order information in a composite holographic lexicon. *Psychological Review*, *114*, 1-37.
- Kieras, D. E., & Meyer, D. E. (1997). An Overview of the EPIC architecture for cognition and performance with application to human-computer interaction. *Human-Computer Interaction*, *12*, 391-438.
- Kohonen, T. (1995). *Self-organizing maps: Vol. 30. Springer Series in Information Sciences*. Berlin: Springer.
- Kosslyn, S. M., Ball, T. M., & Reiser, B. J. (1978) Visual images preserve metric spatial information: Evidence from studies of image scanning. *Journal of Experimental Psychology: Human Perception and Performance*, *4*, 46-60.
- Laird, J., Newell, A., & Rosenbloom, P. S. (1987). Soar: An architecture for general intelligence. *Artificial Intelligence*, *33*, 1-64.

- Laird, J. E., & Rosenbloom, P. S. (1996). The evolution of the soar cognitive architecture. In Steier, D. M., & Mitchell, T. M. (Eds.), *Mind Matters: A tribute to Allen Newell* (pp. 1-50). Mahwah, NJ: Erlbaum.
- Lebiere, C., Gray, R., Salvucci, D., & West, R. L. (2003). Choice and learning under uncertainty: A case study in baseball batting. *Proceedings of the 25<sup>th</sup> Annual Meeting of the Cognitive Science Society*, 704-709.
- Lebiere, C., Wallach, D., & West, R. L. (2000). A memory-based account of the Prisoner's Dilemma and other 2x2 games. *Proceedings of the Third International Conference on Cognitive Modeling*.
- Lebiere, C., & West, R. L. (1999). A dynamic ACT-R model of simple games. *Proceedings of the 21<sup>st</sup> Annual Conference of the Cognitive Science Society*, 296-301. Mahwah, NJ: Erlbaum.
- Linz, P. (1997). *An introduction to formal languages and automata*. Surrey, MA: Jones and Bartlett Publishers.
- Marr, D. (1982). *Vision*. Cambridge, MA: MIT Press.
- Mauro, D. G., LeFevre, J., & Morris, J. (2003). Effects of problem format on division and multiplication performance: Division facts are mediated via multiplication-based representations. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, 29, 163–170.
- McClelland, J. L. (2009). The place of modeling in cognitive science. *Topics in Cognitive Science*, 1, 11-38.
- Miller, G. A. (1956). The magical number seven, plus or minus two: Some limits on our capacity for processing information. *Psychological Review*, 63, 81-97.

- Minsky, M. (1974). A framework for representing knowledge. In J. Haugeland (Ed.), *Mind design II* (pp. 111-142). Cambridge, MA: The MIT Press.
- Monsell, S. (1991). The nature and locus of word frequency effects in reading. In D. Besner & G. W. Humphreys (Eds.), *Basic processes in reading: Visual word recognition*. (pp. 148–197). Hillsdale, NJ: Lawrence Erlbaum.
- Murdock, B. B. Jr. (1982). A theory for the storage and retrieval of item and associative information. *Psychological Review*, *89*, 609-626.
- Murdock, B. B. Jr. (1983). A distributed memory model for serial-order information. *Psychological Review*, *90*, 316-338.
- Murdock, B. B. Jr. (1985). Convolution and matrix systems: A reply to Pike. *Psychological Review*, *92*. 130-132.
- Newell, A. (1973a). You can't play 20 questions with nature and win: Projective comments on the papers of this symposium. In W. G. Chase (Ed.), *Visual Information Processing* (pp. 283-310). New York: Academic Press.
- Newell, A. (1973b). Production systems: Models of control structures. In W. G. Chase (Ed.), *Visual Information Processing* (pp. 463-526). New York: Academic Press.
- Newell, A. (1990). *Unified theories of cognition*. Cambridge, MA: Harvard University Press.
- Newell, A. & Simon, H. (1976). Computer science as empirical enquiry: Symbols and search. In J. Haugeland (Ed.), *Mind Design II* (pp. 81-110). Cambridge, MA: The MIT Press.
- Pazzani, M., & Billsus, D. (2007). Content-based recommendation systems. In P. Brusilovsky, A. Kobsa & W. Nejdl (Eds.), *The adaptive web: Methods and*

*strategies of web personalization* (pp. 325–341). Berlin-Heidelberg, Germany: Springer-Verlag.

- Pavlik, P. I., Jr., & Anderson, J. R. (2005). Practice and forgetting effects on vocabulary memory: An activation-based model of the spacing effect. *Cognitive Science*, *29*, 559-586.
- Pierrakos, D., Paliouras, G., Papatheodorou, C., & Spyropoulos, C. D. (2003). Web usage mining as a tool for Personalization: A survey. *User Modeling and User-Adapted Interaction*, *13*(4), 311–372.
- Pike, R. (1984). Comparison of convolution and matrix distributed memory systems for associative recall and recognition. *Psychological Review*, *91*, 281-294.
- Plate, T. A. (1995). Holographic reduced representations. *IEEE Transactions on Neural Networks*, *6*, 623-641.
- Pool, R. (1995). Putting game theory to the test. *Science*, *267*, 1591-1593.
- Poundstone, W. (1992). *Prisoner's Dilemma*. New York: Doubleday.
- Pribram, K. H. (1971). *Languages of the brain: Experimental paradoxes and principles in neuropsychology*. Englewood Cliffs, NJ: Prentice-Hall.
- Pribram, K. H. (1987). The implicate brain. In B. J. Hiley and F. D. Peat (Eds.). *Quantum implications: Essays in honour of David Bohm* (pp. 365–371). London: Routledge.
- Pribram, K. H. (1991). *Brain and perception: Holonomy and structure in figural processing*. Hillsdale, NJ: Lawrence Erlbaum.

- Pylyshyn, Z. W. (1998). Introduction: Cognitive architecture and the hope for a science of cognition. In Z. W. Pylyshyn (Ed.), *Constraining cognitive theories* (pp. 1-7). Stamford, CT: Ablex Publishing Corporation.
- Pylyshyn, Z. W. (1999). What's in your mind. In E. Lepore & Z. Pylyshyn (Eds.), *What is cognitive science?* (pp. 1-25). Malden, MA: Blackwell
- Radvansky, G. A., Spieler, D. H., & Zacks, R. T. (1993). Mental model organization. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, *19*, 95-114.
- Restle, F. (1966). Run structure and probability learning: Disproof of Restle's model. *Journal of Experimental Psychology*, *72*, 382-389.
- Ritter, F. E., & Wallach, D. P. (1998). Models of two-person games in ACT-R and Soar. *Proceedings of the Second European Conference on Cognitive Modelling*. 202-203. Thrumpton, UK: Nottingham University Press.
- Rizzuto, D. S., & Kahana, M. J. (2001). An auto-associative neural network model of paired-associate learning. *Neural Computation*, *13*, 2075-2092.
- Rose, R. M., & Vitz, P. C. (1966). The role of runs of events in probability learning. *Journal of Experimental Psychology*, *72*, 751-760.
- Rosenblatt, F. (1958). The perceptron: A probabilistic model for information storage and organization in the brain, *Psychological Review*, *65*, 386-408.
- Rosch, E. H. (1975). Cognitive representations of semantic categories. *Journal of Experimental Psychology: General*, *104*(3), 192-233.
- Rumelhart, D. E., Hinton, G. E., & Williams, R. J. (1986). Learning representations by back-propagating errors. *Nature*, *323*, 533-536.

- Rutledge-Taylor, M. F. (2005). Can ACT-R realize “Newell’s Dream”? In B. G. Bara, L. Barsalou & M. Bucciarelli (Eds.) *Proceedings of the 27<sup>th</sup> Annual Conference of the Cognitive Science Society* (pp.1895-1900). Stresa, Italy: University and Polytechnic of Turin.
- Rutledge-Taylor, M. F., Pyke, A. A., & West, R. L. (2010). A holographic model of frequency and interference: Rethinking the problem size effect. Carleton University Cognitive Science Technical Report 2010-02. URL <http://www.carleton.ca/ics/TechReports>
- Rutledge-Taylor, M. F., Pyke, A. A., West, R. L. & Lang, H. (2010). Modeling a three term fan effect. In D. D. Salvucci & G. Gunzelmann (Eds.), *Proceedings of the Tenth International Conference on Cognitive Modeling* (pp. 211-216). Philadelphia, PA: Drexel University.
- Rutledge-Taylor, M. F., Vellino, A. & West, R. L. (2008). A holographic associative memory recommender system. *Proceedings of the Third International Conference on Digital Information Management*, 87-92. London.
- Rutledge-Taylor, M. F., Vellino, A. & West, R. L. (2010). Dynamically structured holographic memory for recommendation. Carleton University Cognitive Science Technical Report 2010-01. URL <http://www.carleton.ca/ics/TechReports>
- Rutledge-Taylor, M. F. & West, R. L. (2004). Cognitive modeling versus game theory: Why cognition matters. In M. Lovett, C. Schunn, C. Lebiere & P. Munro (Eds.), *Proceedings of the Sixth International Conference on Cognitive Modeling* (pp. 255-260). Pittsburgh, PA: Carnegie Mellon University/University of Pittsburgh.

- Rutledge-Taylor, M. F. & West, R. L. (2005). ACT-R versus neural networks in rock=2 paper, rock, scissors. *Proceedings of the Twelfth Annual ACT-R Workshop*, 19-23. Trieste, Italy: Universita degli Studi di Trieste.
- Rutledge-Taylor, M. F. & West, R. L. (2007). MALTA: Enhancing ACT-R with a holographic persistent knowledge store. In D. S. McNamara & J. G. Trafton (Eds.), *Proceedings of the 29<sup>th</sup> Annual Conference of the Cognitive Science Society* (pp. 1433-1438). Nashville, TN: Cognitive Science Society.
- Rutledge-Taylor, M. F. & West, R. L. (2008). Modeling The fan effect using dynamically structured holographic memory. In B. C. Love, K. McRae, & V. M. Sloutsky (Eds.), *Proceedings of the 30<sup>th</sup> Annual Conference of the Cognitive Science Society* (pp. 385-390). Washington, DC: Cognitive Science Society.
- Samuel, A. G. (1981). Phonemic restoration: Insights from a new methodology. *Journal of Experimental Psychology: General*, 110, 474-494.
- Samuelson, L. (1997). *Evolutionary games and equilibrium selection*. Cambridge, MA: MIT Press.
- Sarwar, B., Karypis., G., Konstan, J., & Riedl, J. (2000). Analysis of recommendation algorithms for E-Commerce. *ACM Conference on Electronic Commerce*, 158–167.
- Sarwar, B., Karypis., G., Konstan, J., & Riedl, J. (2001). Item-based collaborative filtering recommendation algorithms. *WWW '01: Proceedings of the 10th International Conference on World Wide Web*, 285–295, New York: ACM Press.
- Schultheis, H., Barkowsky, T., & Bertel, S. (2006). LTM<sup>C</sup> — An improved long-term memory for cognitive architectures. In D. Fum, F. D. Missier, & A. Stocco (Eds.)

*Proceedings of the Seventh International Conference on Cognitive Modeling* (pp. 274-279). Trieste, Italy

- Siegler, R. (1987). The perils of averaging data over strategies: An example from children's addition. *Journal of Experimental Psychology: General*, 116, 250-264.
- Stewart, T. C., Choo, X., & Eliasmith, C. (2010). Dynamic behaviour of a spiking model of action selection in the basal ganglia. In D. D. Salvucci & G. Gunzelmann (Eds.), *Proceedings of the Tenth International Conference on Cognitive Modeling* (pp. 235-240). Philadelphia, PA: Drexel University.
- Stewart, T. C. & Eliasmith, C. (2008). Building production systems with realistic spiking neurons. In B. C. Love, K. McRae, & V. M. Sloutsky (Eds.), *Proceedings of the 30<sup>th</sup> Annual Conference of the Cognitive Science Society* (pp. 1759-1764). Washington, DC.
- Stewart, T. C. & West, R. L. (2007). Deconstructing and reconstructing ACT-R: Exploring the architecture space. *Cognitive Systems Research*, 8, 227-236.
- Taylor, A. G., (2006). Introduction to cataloging and classification. (10th ed.), Westport, CT: Libraries Unlimited.
- Torres, R., McNee, S., Abel, M., Konstan, J., & Riedl, J. (2004). Enhancing digital libraries with TechLens+. *Proceedings of the 2004 Joint ACM/IEEE Conference on Digital Libraries*, 228-236.
- Tulving, E. (1972). Episodic and semantic memory. In E. Tulving and W. Donaldson (Eds.) *Organisation of memory*. London: Academic Press.
- Van Gelder, T. (1998). The dynamical hypothesis in cognitive science. *Behavioral and brain sciences*, 21, 615-628.

- Van Gelder, T. J., & Port, R. (1995). It's about time: An overview of the dynamical approach to cognition. In R. Port & T. van Gelder (Eds.), *Mind as motion: Explorations in the dynamics of cognition*. (pp. 1-43). Cambridge, MA: MIT Press.
- Van Maanen, L., & Van Rijn, H. (2007). An accumulator model of semantic interference. *Cognitive Systems Research, 8*(3), 174-181.
- Vellino, A. & Zeber, D. (2007). A hybrid, multi-dimensional recommender for journal articles in a scientific digital library. *Proceedings of the 2007 IEEE/WIC/ACM International Conferences on Web Intelligence and Intelligent Agent Technology - Workshops*, 111–114.
- Verguts, T. & Fias, W. (2005). Interacting neighbours: A connectionist model of retrieval in single-digit multiplication. *Memory & Cognition, 22*, 1-16.
- Vitz, P. C., & Todd, T. C. (1967). A model of learning for simple repeating binary patterns. *Journal of Experimental Psychology, 75*, 108-117.
- VonNeumann, J., & Morgenstern, O. (1944) *Theory of games and economic behaviour*. Princeton, NJ: Princeton University Press.
- Wagenaar, W. A. (1972). Generation of random sequences by human subjects: A critical survey of the literature. *Psychological Bulletin, 77*, 65-72.
- Ward, L. M. (1973). Use of markov-encoded sequential information in numerical signal detection. *Perception and Psychophysics, 14*, 337-342.
- Ward, L. M., Livingston, J. W., & Li, J. (1988). On probabilistic categorization: The Markovian observer. *Perception and Psychophysics, 43*, 125-136.

- Wang, J., de Vries, A., & Reinders, M. (2006). Unifying user based and item-based collaborative filtering approaches by similarity fusion. *Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, 501–508.
- West, R. L. (1998). Zero sum games as distributed cognitive systems. *Proceedings of the Complex Games Workshop*. Tsukuba, Japan: Electrotechnical Laboratory Machine Inference Group.
- West, R. L., & Lebiere, C. (2001). Simple games as dynamic, coupled systems: Randomness and other emergent properties. *Cognitive Systems Research*, 1(4), 221-239.
- West, R. L., Pyke, A. A., Rutledge-Taylor, M. F. & Lang, H. (2010). Interference and ACT-R: New evidence from the fan effect. In D. D. Salvucci & G. Gunzelmann (Eds.), *Proceedings of the Tenth International Conference on Cognitive Modeling* (pp. 277-281). Philadelphia, PA: Drexel University.
- Winograd, T. (1972). Understanding natural language. *Cognitive Psychology*, 1, 1-191.
- Zatorre, R. (November, 2004). Processing of music and speech by the human auditory cortex: Neuroimaging evidence. *Carleton Cognitive Science Research Seminar*. Ottawa, ON.
- Zbrodoff, N. J. (1995). Why is  $9 + 7$  harder than  $2 + 3$ ? Strength and interference as explanations of the problem-size effect. *Memory and Cognition*, 23(6), 689–700.
- Zbrodoff, N. J. & Logan, G. D. (2005). What everyone finds: The problem-size effect. In J. I. D. Campbell (Ed.), *Handbook of mathematical cognition* (pp. 331–346). New York: Psychology Press.

# DYNAMICALLY STRUCTURED HOLOGRAPHIC MEMORY: THE MECHANISMS

## Abstract

Dynamically Structured Holographic Memory (DSHM), a system for modeling memory, based on Jones and Mewhort's BEAGLE model of the lexicon is presented (Jones & Mewhort, 2007). Unlike BEAGLE, DSHM is not limited to modeling the lexicon. It offers a flexible system for representing knowledge, and can be used to store data in a format similar to that of ACT-R's declarative memory system (Anderson & Lebiere, 1998), ACT-R DM hereafter. A detailed account of the mechanisms internal to DSHM is presented, here. The cognitive interpretation of DSHM is discussed in the companion paper appearing in this portfolio, entitled "Dynamically Structured Holographic Memory: Cognitive Interpretation".

## Introduction

In this paper, the details of how Dynamically Structured Holographic Memory (DSHM) works at a mechanical level are presented. It should be used an authoritative description of the high-level algorithms that underlie the processes that allow DSHM to produce the memory effects that are describe in the other papers contained in this portfolio. Because this paper is meant to be accessible to non-programmers, this paper does not provide all of the fine algorithmic details of the DSHM source code. The source code used by the author to create DSHM is available upon request.

The design of DSHM has been a long gradual process. It has evolved as the number of applications for which it can be used has grown. Nearly all of the formulae presented below have changed at least once since the most basic version of DSHM was

programmed, using python, in 2007. Some of the changes have been as recent as several months prior to the writing of this document. The primary force behind these changes has been pragmatic. Changes to DSHM have always been toward enabling better models of human memory to be created. As such, some of the methods described below may not be intuitive. However, in every case, the solution to a problem began with the simplest, most intuitive, means, and evolved to the current state only after models using the simple method failed.

A primary example of this is the Combine function which reconciles two real valued numbers which independently rate the suitability of a recalled entity in memory, referred to as a Completion. The first function used, merely calculated the product of these values. The idea being that the weakness of a piece of evidence should proportionally discount the recalled Completion.

However, this proved problematic as it made the assumption that no value could be less than 0.0, which is not the case. Additionally, it caused one piece of weak evidence to obliterate another piece of strong evidence, if it was weak enough. For example, if one piece of evidence referred to as the cosine value was high, say 0.85, but the other, referred to as the confidence value was very low, say 0.05, the product would be 0.0425, which is not a very strong endorsement despite the fact that one of the pieces of evidence recommends the Completion quite strongly.

The solution was to design a more nuanced function that treated all positive evidence as strengthening the final value of a Completion, while allowing for circumstances where once or both values is negative (see figure 5).

DSHM is based on BEAGLE (Jones & Mewhort, 2007), a model of the lexicon, but augments its capabilities so as to make it a general purpose system for modeling human memory. As such, the first section of this paper, after this introduction, is a quick review of the mechanisms underlying BEAGLE. This will provide the reader with a basic understanding of how BEAGLE works. The subsequent components of this paper are dedicated to DSHM.

### Beagle

DSHM is based on Jones & Mewhort's (2007) model of the lexicon, BEAGLE (Bound Encoding of the Aggregate Language Environment). The following is a summary of the mechanics of how BEAGLE works. It is presented to as to be clear what features of DSHM are inherited from BEAGLE and which are new. The summary is condensed and therefore somewhat dense. For less compact and complete discussion of BEAGLE see Jones & Mewhort (2007).

### *Information Representation*

BEAGLE is a holographic memory system, related to TODAM (Murdock, 1982; 1983), which stores information in vectors using convolution as a storage mechanism, and correlation as a retrieval mechanism. The BEAGLE system consists of a large set of words, each of which is represented as a pair of vectors. Each vector consists of 2048 real valued elements and has a Euclidean length of 1.0. One vector, called the environmental vector is a static internal representation of the word (an alphanumeric string being the external representation). The other vector, called the memory vector is

dynamic and represents an accumulated history of all of the sentences in which the word has occurred.

It should be emphasised here that the vectors described in this dissertation are vectors consisting of numerical elements, and are not vectors of content bearing items.

The values of the elements of the environmental vector are randomly generated according to an algorithm that ensure that the values are normally distributed around 0.0, and have a variance of  $1/n$ , where  $n$  is the number of elements in the vector. This distribution of values is needed to ensure that the circular correlation operation decodes the circular convolution operation described below. Due to the high number of dimensions, there is less than a 3% chance that the absolute value of the cosine of two random environmental vectors is greater than 0.05, and only approximately a 0.0% chance that it is greater than 0.1. As such, any two random environmental vectors are assumed to be orthogonal.

The values of environmental vectors are not assigned based on the meaning of the items they represent. This is because is the primary function of BEAGLE to discover the semantic and syntactic relationships between items. The meanings of items will be reflected in the memory vectors of the items, once a sufficiently large set of data is computed. See the Comparing Vectors: Computing Association Strengths section of *Dynamically Structured Holographic Memory: Cognitive Interpretation*.

#### *Creating Associations*

When a sentence is imported into the system (i.e., it simulates reading the sentence) a representation for each word in the sentences is created, if none exists already, by generating a random environmental vector and, by default, also initializing

the memory vector to this vector. Then, for each word in the sentence, associations between the word and the other words in the sentences are created. This is done by computing vectors that represent various relationships that the word bears to the other words, and then adding these vectors to the memory vector of the given word.

### *Encoding Co-occurrence*

Two kinds of relationships exist between words. The first is simple co-occurrence. To encode that words have co-occurred in the same sentence, the environmental vector of each word is added to the memory vector of each other word.

For example, if BEAGLE reads the sentence “cats run fast”, the environmental vectors of ‘run’ and ‘fast’ are added to the memory vector of ‘cats’ (see calculation 1, where the ‘=’ symbol is an assignment operator). Here, the addition of two vectors is taken to mean the simple element-wise addition of the vectors; e.g., the first element of the result is equal to the sum of the values of the set consisting of the first element of each of the added vectors. The notation  $A.e$  is used to denote the environmental vector of  $A$ ;  $A.m$  is used to denote the memory vector of  $A$ .

$$(1) \text{ cats.m} = \text{cats.m} + \text{run.e} + \text{fast.e}$$

The memory vectors of run and fast are similarly modified (see calculation 2 and 3).

$$(2) \text{ run.m} = \text{run.m} + \text{cats.e} + \text{fast.e}$$

$$(3) \text{ fast.m} = \text{fast.m} + \text{cats.e} + \text{run.e}$$

As a result, the memory vector of each word will be similar in shape to the environmental vectors of the other words. Thus, if the cosine of one word’s environmental vector and another word’s memory vector is computed, the result will

statistically greater than 0.0 if they have co-occurred together, and will be approximately equal to 0.0 if they have not. Here the cosine of two vectors is defined as the dot product of the two vectors; each vector is normalized to a Euclidean length of 1.0 prior to computing the cosine value.

This co-occurrence encoding does not preserve any information about the relative order of the words. In order to include this information, a second, considerably more complex algorithm is subsequently applied.

#### *Encoding Word Order*

When a sentence is imported into BEAGLE the memory vector of each word is modified with a vector representing the positions of the other words in the sentence, relative to the given word. This is done as follows.

For each word in a sentence, which for a given instance will be referred to as the target word, a copy of the sentence is created with the *target word* replaced by a system defined placeholder,  $\Phi$ . This placeholder can be interpreted as meaning ‘self’. That is, each word learns of the positions of other words relative to itself as ‘self’ and not using its own environmental vector representation.

For every subset of words surrounding (and including) the target word, within a maximum range of three words (on either side), the environmental vectors of the elements of the subset are combined via circular convolution (shown in figure 2; Plate, 1995), and the result is added to the memory vector of the target word. The details of this process will be described using an example.

*Figure 2.* The function defining the circular convolution,  $t$ , of two vectors,  $c$  and  $x$ .

The following example describes the process of importing the sentence 'adventurous birds circle downwards' into BEAGLE. To make the example easier to read, the four words will be referred to by their first letters, in upper-case.

When the sentence "A B C D" is imported, the memory vector of B (to pick an arbitrary example item) is modified as follows. First, B is made sensitive to the fact that it appears with the three other words (i.e., co-occurrence). The environmental vectors of A, C, and D are simply added to the memory vector of B (as in the previous example).

Second, the positions of the other three words, relative to B is encoded, and added to the memory vector of B. The first step in doing this is to create a copy of the sentence from the perspective of the target word (B). This is done by replacing the B term with the system placeholder vector  $\Phi$ , resulting in the sentence "A  $\Phi$  C D". Then, for each subset of contiguous words that include  $\Phi$ , (i.e., 'A  $\Phi$ ', 'A  $\Phi$  C', 'A  $\Phi$  C D', ' $\Phi$  C', ' $\Phi$  C D'), the sequence of these words is encoded and added to the memory vector of B.

The details of how a sequence of items is encoded are now described. In each case, the binding process is iterative, working left to right. A vector representing the left component is encoded as 'left' and a vector representing the right is encoded as 'right'. Encoding a vector as left (or, right) is accomplished by permuting the elements of the vector uniquely according to a predefined pattern. The notation  $L(v)$  is used to denote the encoding of a vector 'v' as left; and  $R(v)$  is used to denote encoding 'v' as right. Once

the left and right components are encoded as such, they are bound together via circular convolution. The symbol  $\odot$  is used to denote the circular convolution operator.

The encoding process starts with a left component consisting of the environmental vector of the first word in the set and the right component starts as the environmental vector of second word. Thus, given the sequence 'A  $\Phi$  C', the process would start by computing calculation 4.

$$(4) L(A.e) \odot R(\Phi.e)$$

On the next iteration, the left component is the result of the initial binding and the right element is the third element of the set as presented in calculation 5.

$$(5) L(L(A.e) \odot R(\Phi.e)) \odot R(C.e)$$

In general, the left component is the accumulated binding of the first n elements in the set and the right is the n+1<sup>th</sup> element. This binding process repeats until the last element has been bound to the rest. The vector representing the final result of binding a set of words together is added to the memory vector of the target word.

The total change to the memory vector of B is presented in calculation 6.

$$\begin{aligned} (6) B.m &= B.m + A.e + C.e + D.e \\ &+ L(A.e) \odot R(\Phi.e) \\ &+ L(L(A.e) \odot R(\Phi.e)) \odot R(C.e) \\ &+ L(L(L(A.e) \odot R(\Phi.e)) \odot R(C.e)) \odot R(D.e) \\ &+ L(\Phi.e) \odot R(C.e) \\ &+ L(L(\Phi.e) \odot R(C.e)) \odot R(D.e). \end{aligned}$$

The process would be applied for each of the words in the sentence read by BEAGLE.

### *Retrieving Associations*

The encoding algorithms described above are reversible, in that from the memory vectors of the words represented in the system, information about which other words a given word has occurred with can be extracted. When decoded, memory vectors are normalized to a Euclidean length of 1.0, prior to any operations performed upon them.

#### *Decoding Co-occurrence*

Simple co-occurrence is discovered by comparing the (normalized) memory vector of a word to the environmental vectors of the other words in the system, via a cosine calculation. A high cosine value indicates that the two words have co-occurred frequently. Two words that have never co-occurred will produce a cosine comparison value of 0.0, on average. Due the high dimensionality of the vectors (2048 elements), the probability of two random vectors being orthogonal is quite high. A simple simulation verifies this. The cosines of 10,000 pairs of random vectors with 2048 real valued elements were computed. The mean cosine was -0.00025 (s.d. 0.02221); and the mean absolute value of the cosines was 0.01766 (s.d. 0.01346).

#### *Decoding Order*

That a word A has occurred immediately to the left of the word B can be extracted from B by generating a probe vector from the memory vector of B that will bear a high cosine comparison value to the environmental vector of any other word that has appeared to the left of B with sufficient frequency (see calculation 7). This probe is generated by first encoding the system placeholder vector,  $\Phi$ , as the right-hand side element, and then correlating (inverse convolution) this vector with the memory vector of B. The inverse of the left-encode process is applied to the result of the correlation (shown in figure 3),

resulting in a probe vector that bears a positive cosine relationship to the environmental vectors of items that have appeared to the left of the word B. The symbol  $\otimes$  is used to denote the circular correlation operator. The notation  $dL(v)$  is used to denote the reversal of the left-encoding of the vector 'v'.

$$(7) p = dL(R(\Phi.e) \otimes B.m)$$

The resulting probe,  $p$ , is approximately equal to the environmental vector of A. The expression, 'approximately equal' is used for two reasons. First, the correlation of a convolution is a lossy decoding, i.e., it does not result in a perfect restoration of the original components (Plate, 1995). For example,  $A.e \otimes (A.e \odot B.e) = B.e + \text{noise}$ . Second, the memory vector of a word can potentially encode thousands of associations, which adds noise to any decoding of a specific piece of information. Nevertheless, despite the noise, decoding via correlation consistently preserves the appropriate relative ranking of matches via cosine comparison. See *Dynamically Structured Holographic Memory: Cognitive Interpretation* for a discussion of noise from a cognitive perspective.

*Figure 3.* The function defining the circular correlation,  $y$ , of two vectors,  $c$  and  $t$ .

### *Memory Vector Comparison*

In the decoding processes described above, memory vectors are decoded to generate probe vectors that bear a similarity to the environmental vectors of other words.

This is considered the primary means to extracting information from the memory vectors of words. However, another method is to compare the memory vectors of words directly.

One of the features of BEAGLE is that the memory vectors of words in a trained system define a multi-dimensional space where the location of a word in the space reflects its semantic content. The synonymy of two words is inversely proportional to the distance between the words in the space. Thus, the cosine of the memory vectors of two words is a measure of their semantic similarity.

#### DSHM (Dynamically Structured Holographic Memory System)

DSHM (Dynamically Structured Holographic Memory System) is based on BEAGLE, but includes numerous modifications that make it a general purpose memory modeling system. DSHM has been programmed in the Python programming language making use of the Numpy package (<http://numpy.scipy.org/>), as well as in Java making use of the Apache Commons Mathematics Library (<http://commons.apache.org/math/>). The author did not have access to the BEAGLE source code and created DSHM based entirely on the descriptions of the BEAGLE algorithms provided in Jones & Mewhort (2007). As such, it is likely that many of the encoding, decoding and matching algorithms have been implemented differently in DSHM as compared to how they are in BEAGLE.

The architecture of DSHM is similar to that of BEAGLE in that items are represented as objects consisting of a pair of vectors: environmental and memory. However, they differ in that each DSHM item has two additional variables, 'items' and 'ordered', which, respectively, store a list of sub-items and a flag indicating whether that

list is ordered or not. A DSHM item is considered to be an ‘atomic’ (or, ‘terminal’) item if it has no sub-items. An item is considered ‘complex’ if it has sub-items.

### *Items*

DSHM items, both atomic and complex, can be represented as strings; this is the default method for importing items into a DSHM system. DSHM reserves the left and right bracket, the colon, and space symbols. Any other string of symbols is interpreted as a label for a DSHM item. Spaces delimit items that bear an unordered relationship to one another, and colons delimit items that bear an ordered relationship to one another. Colons dominate spaces in that in a mixed string of spaces, colons and item labels: the colons are interpreted as forming complex ordered items before spaces are interpreted. Brackets are used to override the default item boundaries. For example, “A B C” is interpreted as an unordered complex item with three atomic sub-items labelled 'A', 'B', and 'C'. The string “A:B C”, is interpreted as an unordered complex item with two sub-items, the first of which is an ordered complex item with two atomic sub-items A and B, the second of which is a atomic item C. The string “A:[B C]”, is interpreted as an ordered complex item with two sub-items, the first of which is A, the second of which is the unordered complex with two sub-items, B and C.

### *Defining an Instance of DSHM*

DSHM systems are defined by creating a DSHM system object and by supplying four global parameters: vector length, uLambda, oLambda, and decay.

The *vector length* parameter must be a number equal to a power of two. It indicates how many elements long the environmental and memory vectors will be.

BEAGLE makes use of a vector length of 2048. This was found to be adequate for storing language information derived from a corpus approximately equal to the amount of text an average undergraduate student may have read in his or her lifetime (Jones & Mewhort, 2007). Vector lengths as small as 128 have been used to model some memory tasks using DSHM (Rutledge-Taylor, Pyke, West & Lang, 2010; Rutledge-Taylor & West, 2008).

The *oLambda* parameter is identical to the *cLambda* variable in BEAGLE. It determines the maximum distance two items can be from one another in the sub-item list of an ordered complex item and still be associated together. BEAGLE made use of a *cLambda* value of 3.

The *uLambda* parameter in DSHM is used when associating the sub-items of unordered complex items together. It determines the maximum size of subsets of sub-items that are bound together. The items from every subset with a size no greater than the *uLambda*+1 value are bound together and influence each other's memory vectors. There is no equivalent parameter in BEAGLE. However, BEAGLE behaves as if it used a *uLambda* parameter value of 1.

The *decay* parameter controls whether every vector added to a memory vector should be given equal weight, or not. Its purpose is to be available to models of memory degradation.

If the decay parameter is set to false, DSHM behaves like BEAGLE: the memory vector or each word is simply the sum of all the vectors that have been added to it, and grows to a length (Euclidean distance from the origin, not number of elements) equal to the squared root of the number of vectors that have been added to it; see figure 4. If

decay is true, the memory vectors are normalized after each vector is added so as to retain a Euclidean length of 1.0. This compresses the memory vector in such a way that the overall contribution to the shape of the memory vector, by another vector is inversely proportional to the number of subsequent vectors that have been added.

With the decay parameter set to true, the expected cosine correlation between the memory vector of a word A, and the environmental vector of a word B, is  $1 / 2^{(x+1)/2}$ , where  $x$  is the number of vectors that have been added to A.m since B.e was added to A.m; or  $1 / 2^{x/2}$ , if  $x$  includes the addition of B.e. If B.e has never been added to A.m, the value of  $x$  is infinite. If the decay parameter is set to false, the expected cosine of A.m and B.e is  $1 / (x + 2)^{1/2}$ , where  $x$  is the number of vectors that have been added to A.m since B.e was added to A.m; or  $1 / (x + 1)^{1/2}$ , if  $x$  includes the addition of B.e.

$$\text{Cosine} = 1 / (x + 2)^{1/2}$$

*Figure 4.* The formula for calculating the expected cosine correlation between the memory vector of a word A, and the environmental vector of a word B, where  $x$  is the number of vectors that have been added to A.m since B.e was added to A.m and the decay parameter is set to false.

For example, if the words A and B have not yet been associated, the cosine correlation of A.m and B.e will be approximately 0.0 (i.e.,  $1 / 2^\infty$ ). If B.e is added to A.m, the cosine correlation of A.m and B.e will be approximately 0.707 (i.e.,  $1 / 2^{1/2}$ ). This is a high value because 50% of the information in A.m now derives from B.e. After three

random vectors are added to A.m, the cosine correlation of A.m and B.e will fall to approximately 0.250 (i.e.,  $1 / 2^{(3+1)/2}$ ), with decay set to true, and approximately 0.447 with decay set to false.

Table 2 summarizes the results of the process described above, using vector lengths of 128, and up to 64 vector additions. Each cell represents the cosine of A.m and B.e after the number of vectors listed under ‘Additions’ have been added to A.m, and with the decay parameter set to either true or false. The number of additions includes the addition of B.e. Thus, the first column represents the cosine of A.m and B.e immediately after B.e is added to A.m, the second column represents the cosine after 3 more random vectors have been added to A.m, etc. The values are the averages of 1000 simulations.

*Table 2.* Decay parameter comparison

Additions	Decay	
	False	True
1	0.706	0.707
4	0.446	0.249
16	0.242	0.003
64	0.124	-0.001

Thus, decay, using the decay parameter, in DSHM is quite quick. After only 16 total vectors have been added to A.m, the cosine correlation of A.m and B.e is almost zero with decay set to true. In contrast, the cosine value is close to the mathematically predicted value of 0.125, after 64 additions and decay set to false. For this reason, the

decay parameter has not been found to aid in creating models of human memory degradation. The preferred means for accounting for memory degradation is discussed in *Dynamically Structured Holographic Memory: Cognitive Interpretation*.

### *Encoding Associations*

Encoding and decoding in DSHM operates similarly to that of BEAGLE. Ordered lists of terminal items (e.g., the sub-items of an ordered complex item) are encoded almost identically to how sentences are encoded in BEAGLE. The difference is that DSHM does not imbue each item with simple co-occurrence information (by adding the environmental vector of each item to the memory vector of each other item), as does BEAGLE. The reason for this is that this manner of encoding the co-occurrence of items is reserved for unordered items only.

For example, when the complex item “[A:B:C:D]” is imported in to a DSHM system, with an  $\lambda$  of 2, the memory vector of A would be adjusted according to calculation 8.

$$(8) \ A.m = A.m + L(\Phi.e) \odot R(B.e) \\ + L(L(\Phi.e) \odot R(B.e)) \odot R(C.e)$$

Unordered complex items are encoded as follows. First, a list composed of every subset of the sub-items of the unordered item with  $\lambda+1$  or fewer elements is created. Then, the items in each subset in the list are informed of their co-occurrence.

For each item in each subset, the environmental vectors of the other items are convolved together, and the result is added to memory vector of the item.

For example, if the complex item “[A B C D]” is imported in to a DSHM system, with a  $u\Lambda$  of 2, the list of subsets to be associated together would be: [A B C], [A B D], [A C D], [B C D], [A B], [A C], [A D], [B C], [B D], [C D]. In the case of the first of these subsets [A B C], the convolution of the environmental vectors of B and C would be added to the memory vector of A; the convolution of the environmental vectors of A and C would be added to the memory vector of B; and the convolution of the environmental vectors of A and B would be added to the memory vector of C.

The total adjustment to the memory vector of A is presented in calculation 9.

$$\begin{aligned}
 (9) \quad A.m &= A.m + B.e + C.e + D.e \\
 &+ B.e \odot C.e \\
 &+ B.e \odot D.e \\
 &+ C.e \odot D.e
 \end{aligned}$$

The memory vectors of the other words would be similarly adjusted. The effect of adding the convolved sets of items’ environmental vectors is to make the modified memory vector sensitive to the co-occurrence of particular combinations of other items and not just those other items individually. That is, if A was modified by adding the unconvolved environmental vectors of the other items only (i.e.,  $A.m = A.m + B.e + C.e + D.e$ ), there would be no way of determining whether A had occurred with more than one of the other items at the same time, or with only the individual items alone in separate instances. That is, the effect on A of the item “[A B C D]” being imported into a DSHM model would be indistinguishable from the combined effect of importing the items “[A B]”, “[A C]”, and “[A D]”. With a  $u\Lambda$  of 2, A contains information about which pairs of items it co-occurred with (i.e., A co-occurred with both B and C at the

same time, etc). However, A is not sensitive to the fact that it co-occurred with B, C and D at the same time. For this to be the case the system would need a  $u\Lambda$  value of three (or greater), so as to include  $(B.e \odot C.e \odot D.e)$  in the calculation of the modification of A.m.

For a more detailed example of this process see the Hierarchical Completion example in the appendix.

### *Compositional Structure*

Items with compositional structure are encoded recursively. That is, when an item with complex sub-items is imported into DSHM, associations between the complex sub-items' items are computed as well. For example, if the string "A:[B C]" is imported into DSHM, the associations between B and C are computed in addition to the associations between A and [B C]. There are no direct associations between items and the sub-items of a neighbouring item. In the case just presented, A and B and are not associated with one another. However, sub-items in all positions of a hierarchically structured complex item influence the information decoding process.

### *Decoding Associations*

The memory vectors of items in a DSHM system can be decoded to determine which items they co-occurred with. This is done via pattern completion. Here it is necessary to define some terms.

### *Definitions*

A query item is an unspecified item that acts like a variable. It is represented by a string preceded with a question mark, e.g., '?x'. A query item is an atomic incomplete item.

Items were described above as falling into two categories: atomic and complex. An additional distinction is made between two types of items: complete items, and incomplete items. A complete item is a complex item with all of its sub-items fully specified. Atomic items (except query items) are considered complete items.

An incomplete item is defined recursively. A complex item with a query item as a sub-item is an incomplete item (e.g., [A ?x]). A complex item with a sub-item that is an incomplete item is an incomplete item (e.g., [A [B ?x]]). Query items are considered incomplete items.

A Completion is an object that contains both a complete item and a confidence value. The item corresponds to an incomplete item with complete items substituted for all of the query items in the incomplete item. The confidence value quantifies how strongly the complete item completes the incomplete item. How the confidence value of the completion is computed is discussed below in the Completion Values section. Each Completion also keeps track of separate confidence values for each of the items substituted for each of the query items that exist in the original incomplete item. Completions will be written in text in the format (item, value); for example, ([B C], 0.716).

A Partial Completion is a Completion with complete items substituted for some query items, but not all query items. For example, ([[A1 A2 A3]:[?y B2 B3]], 0.723) is a partial completion of an initial incomplete item [[A1 A2 ?x]:[?y B2 B3]].

In the following accounts of the process of completing incomplete items, it is convenient to use the term ‘completion’ for two subtly different referents. As a convention, the term ‘Completion’, with a capital ‘C’, will be used to refer to a data object, defined here, which contains a complete item, and a confidence value. A Completion is not a kind of item. A Completion contains an item and a confidence value.

The term ‘completion’, with a lower-case ‘c’, can be used to refer to the item contained in a Completion. For example, “... *only atomic items are permissible completions of query items*” should be interpreted as meaning, “... *only atomic items are permissible as the items contained in the Completions of query items*”. Thus, when used to refer to a complete item, ‘completion’ only refers to a complete item that corresponds to an incomplete item with complete items substituted for its query items.

Additionally, ‘completion’ is also used in the grammatical sense as the act of completing an incomplete item.

The uses of the terms ‘Completion’ and ‘completion’ may be the source of some confusion at first. However, once the meanings of these terms are well understood, using them allows the text in which they appear to flow more smoothly.

The complete sub-items of an incomplete item are referred to as context items. This is because they provide the context for determining what the substitutions for the incomplete sub-items should be.

### *Completing an Incomplete Item*

Associations are decoded by presenting an incomplete item to a DSHM model for completion. The sources of information for completing the item are the context items (i.e., the complete sub-items) of the incomplete item. The result of item completion is a rank ordered list of candidate Completions.

Unlike the encoding process which is parameter free, the decoding process accepts five parameters: *atomicOnly*, *bestOnly*, *mem*, *gap*, and *oneProbe*.

The *atomicOnly* parameter determines whether only atomic items are permissible as completions (i.e., items contained in Completions) of query items. If the value is true, only atomic items are permissible. If the value is false, both atomic and complex items are permissible completions of query items. The value is set to false by default. However, it can be set to true if it is known in advance that complex items are not appropriate completions of the query items for the given retrieval.

If the *bestOnly* parameter is set to true, all lists of Completions after the top candidate are truncated. This parameter is applied recursively; i.e., Completions of sub-items are also truncated. The *bestOnly* parameter is set to false by default. It should be set to true for testing purposes only.

By default, incomplete item completion is based only on the generation of probes that match the environmental vectors of candidate items, as discussed above. The *mem* parameter determines whether probe vectors designed to match the memory vectors of items should be generated and used as well. If set to true both memory vector probes and environmental probes are used. If set to false only environmental vector probes are used. The *mem* parameter is set to false by default.

The *gap* parameter determines whether the list of Completions is truncated after the first cluster of Completions, starting with the first completion. This cluster is defined according to an algorithm that searches for the greatest difference, referred to as the gap, in the natural logarithms of confidence values of the Completions (which are ordered largest to smallest). This gap delineates the items that belong to the first cluster of items and those that do not.

This parameter is set to false by default for legacy purposes. Setting it to true is considered standard practice, as it is necessary for the DSHM theory of recall. See the Gap Example in the appendix.

As is the case with BEAGLE, decoding involves the generating of probe vectors used for testing potential candidate completions of incomplete items. When more than one context item is present, the opportunity to generate several probes exists (see the examples below). The *oneProbe* parameter determines whether sets of probe vectors are combined to create a single probe representing the average of the available probes, or not, prior to testing potential matches. If the parameter is set to false, the score for the each candidate substitution is the average cosine value of the item and each of the probes. If the parameter is set to true, the probes are averaged to produce a single probe vector. In this case, the score for each candidate substitution is the cosine value of the item and this single aggregate probe.

The *oneProbe* parameter is set to false, by default. See the OneProbe Example in the appendix for an explanation of why better results are achieved with this setting.

### *Recursive Completion*

The completion process is recursive. The base case is when the item to complete is a query item. Since there are no complete items to use as the basis for evaluating the query item, no possible completion can receive any support. Thus, every item in the system is an equally good candidate. The list of Completions of the query item consists in one Completion for every item in the system.

When the item to complete, called the root item, has a mix of complete and incomplete sub-items, the incomplete items are completed first. Next, the complete sub-items are decoded to produce a set of probe vectors that represent the knowledge residing in each of the complete items about what other items, in the positions of the incomplete items, have co-occurred with them.

These probe vectors are used to evaluate the degree of association between each of the candidate completions of the incomplete sub-items and the complete sub-items. A new list of Completions of the root item is compiled. The ranking of the Completions in this list is based on an algorithm that combines the values of the Completions of the sub-items, the values given to the degree to which the items in the completions are associated with the context items, and values assigned to the context items. The mechanism for computing Completion values is described below in the Completion Values section.

### *Generating Probes*

Probe vectors are generated in a manner analogous to how they are produced in BEAGLE. As is the case in BEAGLE, all of the encoding operations in DSHM are reversible.

Given a unordered incomplete item  $[A B ?x]$ , four probe vectors would be produced (provided the  $u\lambda$  parameter is equal to 2 or more), according to calculations 10 to 13:

$$(10) \quad pAe = A.m$$

$$(11) \quad pBe = B.m$$

$$(12) \quad pABe = B.e \otimes A.m$$

$$(13) \quad pBAe = A.e \otimes B.m$$

The label for a probe follows the convention of starting with a lower-case 'p', and ending with a lower-case 'e' if it is designed to match the environmental vector of potential completions, and a lower-case 'm' is designed to match the memory vector of potential completions. The first of the infix capital letters indicates from which context item's memory vectors was used as the source of information. Subsequent letters indicate other context items that were used in decoding the memory vector of the first context item.

Given an ordered incomplete item  $[A:B:?x]$ , three probe vectors would be produced (provided the  $o\lambda$  parameter is equal to 2 or more), according to calculations 14 to 16:

$$(14) \quad pABe = dR(L(B.e) \otimes (L(\Phi.e) \otimes A.m))$$

$$(15) \quad pBe = dR(L(\Phi.e) \otimes B.m)$$

$$(16) \quad pBAe = dR(L(\Phi.e) \otimes dR(L(A.e) \otimes B.m))$$

### *Completion Values*

The value of a Completion is equal to the average of values assigned to the sub-items of the item to which it corresponds. The completion (i.e., item) corresponds to an incomplete item with complete items substituted for each of the incomplete items within it. What value to assign a sub-item depends on what sort of item it was in the corresponding incomplete item.

The value of a *context items* is the output of an algorithm that computes the resonance of the item (see the *Resonance* section below). Here, resonance is defined as the degree to which the sub-items of an item are associated together. If X and Y co-occur frequently [X Y] would likely resonate with a high value. If not, the resonance value could be low. The resonance of a complete atomic item is 1.0.

The Completion of a *query item* has a value of 0.0.

The value of the Completion of an *incomplete item* is not the value contained in the Completion. The value contained in the Completion quantifies how good of a completion it is based on its constituent parts. For example, if a DSHM model is prompted to complete [B ?x], a Completion ([B C], 0.716) should be interpreted as meaning only that C is highly associated with B.

However, if a DSHM model is prompted to complete [A:[B ?x]], it will first complete the incomplete sub-item [B ?x], producing a Completion ([B C], 0.716). In this case it is important to understand that the value of the [B C] completion, 0.716, does not contain any information at all about how well [B C] completes [A:[B ?x]] in the context of A. A and [B C] could be very highly associated (in an ordered relationship) or not associated at all. No information about this relationship has been determined yet.

Thus, the value of an incomplete sub-item is the result of a function, referred to as the *Combination function*, which takes two parameters:

- 1) the Completion value of the completion of the sub-item, and
- 2) a measure of the degree to which the sibling context items and the completion are associated.

The second parameter is the result of comparing the appropriate vectors of the completion and the probes generated from the context items. For example, if we continue the example begun above, A is decoded to produce the probe pAe. In this case,  $pAe = dR(L(\Phi.e) \otimes A.m)$ .

The cosine of pAe and the environmental vector of [B C], i.e., [B C].e, is a measure of the degree to which A and [B C] are associated. This is shown in calculation 17.

$$(17) \text{ Cosine}(pAe, [B C].e) = 0.335$$

To quantify the overall value of [B C] as completion of [B ?x] relative to A, The value of the completion of [B ?x], 0.716, and the cosine value are fed into the Combine function, defined in Figure 5. The equation differs depending on the valence of the cos (cosine) and conf (confidence) parameters.

This function produces a value that is greater than either of the contributing values, for positive, non-zero, parameters. This is because the range of possible values is [-1.0, 1.0], and two positive values indicate greater net support for the completion in question than either value on its own.

<pre> IF (cos &gt; 0.0 &amp; conf &gt; 0.0)      Combine(cos, conf) = 1 - (1 - cos) * (1 - conf)  ELSE IF (cos &lt; 0.0 &amp; conf &lt; 0.0)      Combine(cos, conf) = (1 + cos) * (1 + conf) - 1)  ELSE      Combine(cos, conf) = cos + conf </pre>
--

*Figure 5.* Combine equation.

The result of calculation 17 shows that the value of [B C], relative to the lone context item A, is 0.820. The value of A is 1.0, by definition. As stated above, the value of a Completion is the mean of the values of the sub-items of the item in the Completion, as described above. Thus, the overall value of the completion [A:[B C]] is  $(1.0 + 0.820) / 2 = 0.910$ .

See the Hierarchical Completion Example in the appendix to see an example of the completion of [A:[B ?x]] that includes all of the lists of completions at each stage in the completion process.

### *Resonance*

Instead of making use of a concept such as base-level activation, as is found in ACT-R, DSHM can compute the resonance of a complex item. This resonance value is a measure of the association strengths between the sub-items of the item.

The algorithm for computing resonance of an item, I, is as follows. For each sub-item,  $S_j$ , of I, an incomplete item,  $N_j$ , is created by replacing  $S_j$  with a query item  $Q_j$ .  $N_j$

is completed and the value of the (correct) Completion where  $S_j$  is substituted for  $Q_j$ , is recorded. The average of the recorded values is the resonance value of the item  $I$ .

### *Canonical Values*

There are two sets of parameters that can be set in DSHM: 1) a set of global parameters that are set when an instance of DSHM is created, and 2) a set of parameters that are set when generating completions. No one single combination of parameter values has proven to work for all DSHM models. However, there are parameter values which work more often than other values. These are presented in tables 3 and 5:

*Table 3.* Default and recommended global parameters

#### Global Parameters

Parameter	Default	Recommended range
vector length	128	128 to 2048
uLambda	2	1 to 3
oLambda	3	1 to 4
decay	false	false

A vector length of 128 should be appropriate for tasks where a relatively low number of unique associations per item are made (e.g., Rutledge-Taylor, Pyke, West & Lang, 2010; Rutledge-Taylor & West, 2008). If the model is required to assimilate a very large number of facts and each item may need to accommodate hundreds of unique associations, a vector length as high as 2048 may be needed (e.g., Rutledge-Taylor,

Vellino & West, 2010). If the model needs to distinguish between subtly different confidence values, higher vector lengths may be required (e.g., *Using DSHM to Model Paper, Rock, Scissors*).

There are no firm rules about what uLambda and oLambda parameter values are permissible. However, there is an underlying assumption that they be constrained by various limits on cognitive capacity. In the case of uLambda it is hypothesized that there is not an unlimited capacity for the number of combinations of sub-items that can be bound together. The number of associations that must be computed increases dramatically with the value of the uLambda parameter and number of sub-items per complex item. Therefore, it is recommended that uLambda be limited in certain scenarios. As a rule of thumb, the total number of associations computed per item should not exceed 100; see table 4. If the mean number of sub-items per item is five or fewer, any uLambda value is permissible. If the mean number of sub-items per item is six, uLambda should not exceed 2. If the mean number of sub-items per item is seven or more, only a uLambda value of 1 should be used.

The number of associations performed when ordered complex items are reinforced is much less than that for unordered complex items. Therefore, the limits on the oLambda parameter are entirely theoretical. In BEAGLE, the value for the equivalent parameter (cLambda), was set to three. This would result in a maximum sequence length of seven words to be associated together. This complies with Miller's magic number of seven, plus or minus two (Miller, 1956). For this same reason, it is recommended that an oLambda value of three be used unless there is a compelling reason to increase this parameter value to four (or more).

*Table 4.* The number of associations that are computed when an unordered complex item is presented to a DSHM model.

Sub-items	uLambda			
	1	2	3	4
2	2	2	2	2
3	6	9	9	6
4	12	24	28	28
5	20	50	70	75
6	30	90	150	180
7	42	147	287	392

As discussed above, the decay parameter should only be set to false.

#### *Computational Cost*

DSHM models are computationally costly to run relative to models generated using other modeling systems. This is because each memory process in a DSHM model requires that many computations be performed on large vectors of real valued numbers.

The temporal cost of vector addition and cosine computation is linearly related to the dimension  $n$  of the vectors under consideration. That is, the time to complete these calculations is  $O(n)$ . The convolution and correlation algorithms discussed above run in  $O(n^2)$  time. However, convolution and correlation computations can be accurately approximated in  $O(n \log n)$  time by making use of Fast Fourier transforms (Plate, 1995).

The computer programs that implement DSHM use this, less expensive, method of computing vector convolutions and correlations.

It should also be noted that generation of item completions requires that probes be compared to every item in the system. Thus, the total cost of generating a completion is  $O(m*n\log n)$ , where the system has  $m$  items, and vectors of length  $n$ .

*Table 5. Default and recommended completions parameter values*

Completion Parameters		
Parameter	Default	Recommended values
atomicOnly	false	true or false
bestOnly	false	false
mem	false	false
gap	false	true
oneProbe	false	false

### *Cognitive Interpretation*

In this paper, DSHM is described at a mechanical level only, and has not included a discussion of how the inputs and outputs of DSHM should be interpreted cognitively. Rather than mix low level mechanics, with high-level theory, the cognitive interpretation of DSHM is discussed in the companion paper, entitled “Dynamically Structured Holographic Memory: Cognitive Interpretation”.

## Appendix

This section includes examples that illustrate various parameter settings.

### *Gap Example*

In Rutledge-Taylor, Pyke, West and Lang (2010), a three term fan effect experiment was run, and modeled using DSHM. Figure 6 shows the confidence values for the top 10 candidate completions of “[green:?x:garage]”. A large drop-off in the values after the first can be observed. This is verified in Figure 7. In this figure the value above each completion index is the difference between the natural logarithm of the confidence value of the completion and the logarithm of the confidence value of the previous completion (i.e., the completion with the closest, higher value). Here, the peak above completion 2 indicates that the greatest difference in the natural logarithm of the confidence values was between the 1<sup>st</sup> and 2<sup>nd</sup> completions. This means that the list is truncated after the 1<sup>st</sup> completion. Thus, the model is reporting that there was only one sentence about a green thing in the garage in the set of study sentences (it was “The green hat is in the garage).

Figure 8 shows the confidence values for the top 10 candidate completions of “[black:mug:?x]”. In this case the position of the largest drop-off is less obvious. However, when the logarithms are calculated, the peak difference is unambiguously above the 4<sup>th</sup> completion, in Figure 9. This means that the list is truncated after the 3<sup>rd</sup> completion. In this case, the model is reporting that there are three sentences about a black mug in the set of study sentences (“The black mug is in the bedroom”, “The black mug is in the yard”, and “The black mug is in the hallway”). In fact, there was only one such sentence in the study list, “The black mug is in the bedroom”. However, the

participant's task was to fill in the blank ("the black mug was in the \_\_\_\_\_"). Therefore, the model's response is interpreted as believing that there are three possible answers. We assumed that the model would fill in the blank with the location word from the top ranked completion, in a manner similar to how ACT-R DM selects the most active chunk during retrieval. In this case it would be the correct word, "bedroom". In Rutledge-Taylor and West (2008), this multiple answer feature of DSHM was used to model participants of Anderson's classic fan effect experiment providing more than one answer to questions, such as "where are the hippies?" (Anderson, 1974).

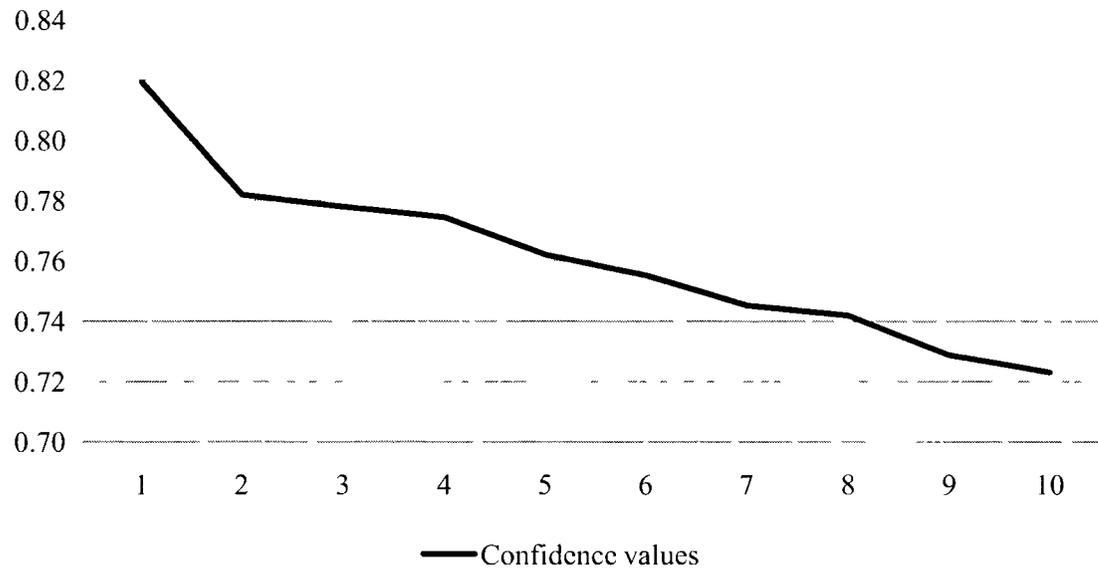


Figure 6. Confidence values for the top 10 completions of [green:?x:garage].

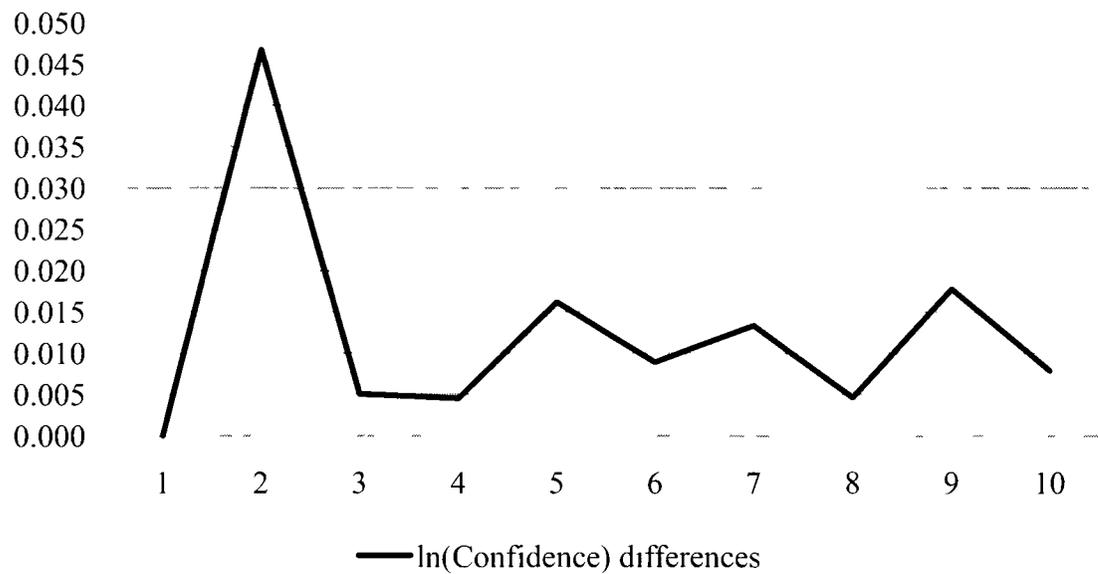


Figure 7. Differences in the natural logarithms of the completion values of [green:?x:garage].

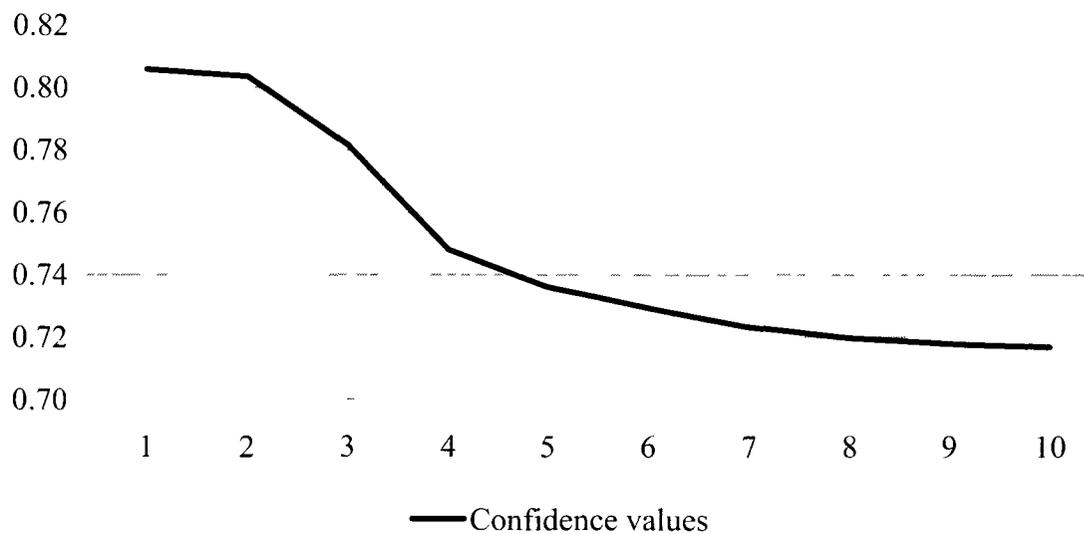


Figure 8. Confidence values for the top 10 completions of [black:mug:?x].

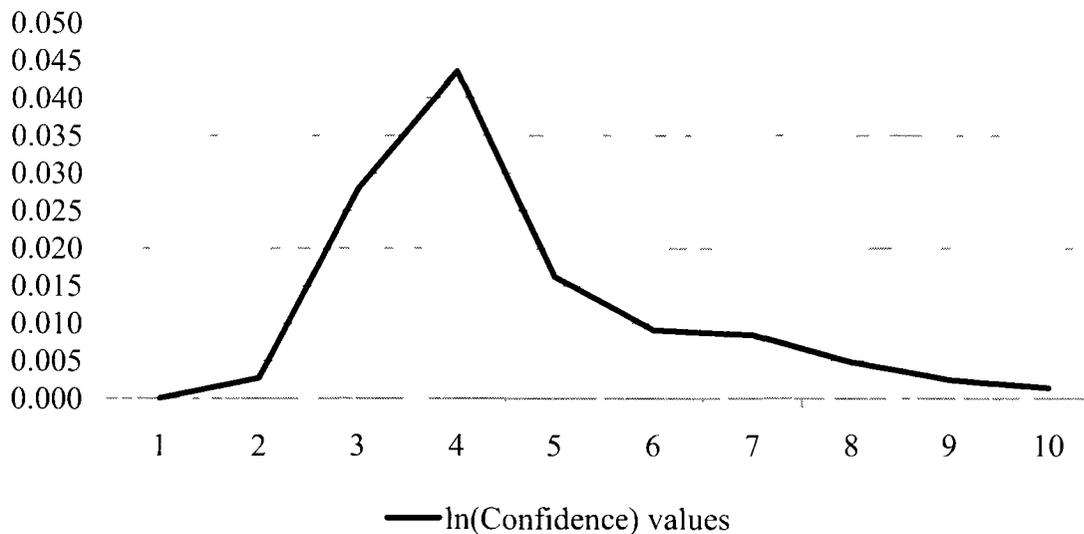


Figure 9. Differences in the natural logarithms of the completion values of [black:mug:?x].

### *OneProbe Example*

In this example, DSHM models were created (vector length=128, uLambda=3, oLambda=1, and decay=false), and trained on the following items:

- [A B C D]
- [A M N O]
- [B P Q R]
- [C S T U]
- [D V W X]
- [E F G H]
- [I J K L]

Each instance of the model was prompted to complete the incomplete item [A B C ?x]. If we consult the list of items memorized above, we should expect that the correct substitution for ?x is 'D'. We ought to expect that if the letters in this list of items were replaced with arbitrary words, to make a list of study sentences, a focused experimental participant ought to recall the correct missing word, most of the time.

To test the effect of setting oneProbe to true or false, 1000 DSHM models completed the incomplete item with parameter settings: atomicOnly=false, bestOnly=false, mem=false, gap=true, and oneProbe=true. An additional 1000 models completed the incomplete item with parameter settings: atomicOnly=false, bestOnly=false, mem=false, gap=true, and oneProbe=false.

In the 'true' cases (i.e., oneProbe=true), the top ranked Completion matched the expected answer of [A B C D] only 17.0% of the time. The mean Completion value of

the top ranked Completion was 0.841, for correct answers. The mean initial cluster size was 3.71.

In the 'false' cases, the top ranked Completion matched the expect answer of [A B C D] 99.4% of the time. The mean Completion value of the top ranked Completion was 0.808, for correct answers. The mean initial cluster size was 1.14.

Thus, the DSHM models in this example produced more accurate results with oneProbe equal to false. Since DSHM is a model of human memory, it is not necessarily the case that greater accuracy makes a better model of human performance. However, it is the case that setting the oneProbe parameter to false does indeed produce models that produce data that fits human data better than models with the oneProbe parameter set to true.

#### *Hierarchical Completion Example*

In this example, a DSHM model (vector length=2048, uLambda=2, oLambda=2, and decay=false), and trained on the following items:

- [A:[B C]]
- [A:[D E]]
- [A:[F G]]
- [B D]
- [B E]
- [B F]

The model was prompted to complete the incomplete item A:[B ?x]] with parameter settings: atomicOnly=true, bestOnly=false, mem=false, gap=true, and oneProbe=false.

Step 1 is for the model to examine the sub-items of A:[B ?x]], which are the context item A and the incomplete item [B ?x]. Step 2 is to recursively complete [B ?x]. Step 3 is to examine the sub-items of [B ?x], which are the context item B and the incomplete item ?x. Step 4 is to recursively complete ?x. Since ?x is a query item, the set of Completions returned from this step are displayed in table 6.

There is no evidence for or against each of these completions of ?x, therefore they each have a value of 0.0. Step 5 is to evaluate each of these completions in the context of the context item B. The probe pBe, which is equal to B.m, is compared to the environmental vector of each of the completions in the list. The results of this comparison are presented in table 7.

*Table 6. Completions of ?x*

Rank	Item	Value
1	A	0.0
2	B	0.0
3	C	0.0
4	D	0.0
5	E	0.0
6	F	0.0
7	G	0.0

*Table 7. Cosines of pBe (B.m) and the environmental vectors of the completions*

Probe	Environmental vector	Cosine
B.m	A.e	0.010
B.m	B.e	0.009
B.m	C.e	0.433
B.m	D.e	0.454
B.m	E.e	0.458
B.m	F.e	0.428
B.m	G.e	0.014

These cosine values are combined with the values of the Completions (in Table 8), to produce the values for the completions of ?x, relative to B.

Since, the combination function outputs x, when given a cosine parameter of x and confidence value of 0.0, as is the case here the values of the completions of ?x are identical to the cosine values calculated.

Step 6 is to generate the set of Completions of [B ?x]. Since, B has a value of 1.0, the values of the Completions will be the average of 1.0 and the relative value, for each completion. These values are shown in table 9. Note that with the gap parameter set to true, Completions [B A], [B B], [B G] are discarded. It is not a coincidence that only Completions corresponding to items the model has been exposed to previously remain.

*Table 8.* Relative values of the completions of ?x to the context item B

Completion	Confidence value	Cosine	Relative value
A	0.000	0.010	0.010
B	0.000	0.009	0.009
C	0.000	0.433	0.433
D	0.000	0.454	0.454
E	0.000	0.458	0.458
F	0.000	0.428	0.428
G	0.000	0.014	0.014

Step 7 is to evaluate each of the completions of [B ?x] in the context of the context item A. Table 10 shows the completions of [B ?x], the confidence values reported in table 9, the cosine of the probe pAe and the environmental vector of the completion, and the relative value of the completion relative to the context time A, as computed by the Combine function.

*Table 9.* Items and values for Completions of [B ?x]

Rank	Item	Value
1	[B E]	0.729
2	[B D]	0.727
3	[B C]	0.716
4	[B C]	0.714

*Table 10.* The relative values of the completions of [B ?x] relative to the context item A

Completion	Confidence value	Cosine	Relative value
[B C]	0.716	0.335	0.82
[B E]	0.729	0.022	0.733
[B D]	0.727	0.059	0.733
[B F]	0.714	0.028	0.722

It should be noted here, that the high cosine of pAe and [B C].e causes the relative value of [B C] to surpass that of the other Completions.

Step 8 is to generate the set of Completions of [A:[B ?x]]. Since, A has a value of 1.0, the values of the Completions will be the average of 1.0 and the relative value, for each completion. This set of Completions consists only in the lone Completions ([A:[B C]], 0.910). Due to the gap parameter being set to true, the other three possible completions ([A:[B E]], 0.867), ([A:[B D]], 0.867), and ([A:[B F]], 0.861) are truncated. Thus, the only completion of [A:[B ?x]] supplied by the model is [A:[B C]]. From a cognitive perspective, this is interpreted as the model judging that there was only one suitable completion.

The relatively high values of the rejected Completions should be interpreted as meaning that there were aspects of those completions which made them seem like good candidates. In this case, each of the rejected candidates included a sub-item that the model had seen before. Thus, the model may have been tempted to judge them as suitable completions of the target incomplete item. However, the fact that there was one lone superior candidate caused the model to reject the other candidates.

## References

- Anderson, J. R. (1974). Retrieval of propositional information from long-term memory. *Cognitive Psychology*, 6, 451-474.
- Anderson, J. R., & Lebiere, C. (Eds.) (1998). *The atomic components of thought*. Mahwah, NJ: Lawrence Erlbaum.
- Jones, M. N., & Mewhort, D. J. K. (2007). Representing word meaning and Order information in a composite holographic lexicon. *Psychological Review*, 114, 1-37.
- Murdock, B. B. Jr. (1982). A theory for the storage and retrieval of item and associative information. *Psychological Review*, 89, 609-626.
- Murdock, B. B. Jr. (1983). A distributed memory model for serial-order information. *Psychological Review*, 90, 316-338.
- Plate, T. A. (1995). Holographic reduced representations. *IEEE Transactions on Neural Networks*, 6, 623-641.
- Rutledge-Taylor, M. F., Pyke, A. A., West, R. L. & Lang, H. (2010). Modeling a three term fan effect. In D. D. Salvucci & G. Gunzelmann (Eds.), *Proceedings of the Tenth International Conference on Cognitive Modeling* (pp. 211-216). Philadelphia, PA: Drexel University.
- Rutledge-Taylor, M. F., Vellino, A. & West, R. L. (2010). Dynamically structured holographic memory for recommendation. Carleton University Cognitive Science Technical Report 2010-01. URL <http://www.carleton.ca/ics/TechReports>
- Rutledge-Taylor, M. F. & West, R. L. (2007). MALTA: Enhancing ACT-R with a holographic persistent knowledge store. In D. S. McNamara & J. G. Trafton

(Eds.), *Proceedings of the 29<sup>th</sup> Annual Conference of the Cognitive Science Society* (pp. 1433-1438). Nashville, TN: Cognitive Science Society.

Rutledge-Taylor, M. F. & West, R. L. (2008). Modeling The fan effect using dynamically structured holographic memory. In B. C. Love, K. McRae, & V. M. Sloutsky (Eds.), *Proceedings of the 30<sup>th</sup> Annual Conference of the Cognitive Science Society* (pp. 385-390). Washington, DC: Cognitive Science Society.

# DYNAMICALLY STRUCTURED HOLOGRAPHIC MEMORY: COGNITIVE INTERPRETATION

## Abstract

Dynamically Structured Holographic Memory (DSHM), a system for modeling memory, based on Jones and Mewhort's BEAGLE model of the lexicon is presented (Jones & Mewhort, 2007). Unlike BEAGLE, DSHM is not limited to modeling the lexicon. It offers a flexible system for representing knowledge, and can be used to store data in a format similar to that of ACT-R's declarative memory system (Rutledge-Taylor & West, 2007). The cognitive interpretation of DSHM is discussed in this paper. See the companion paper, entitled “Dynamically Structured Holographic Memory: The Mechanisms”, for the mechanical details of how DSHM works.

## Introduction

In this paper, the psychological theory underlying Dynamically Structured Holographic Memory (DSHM) is presented. This discussion is relatively abstract, as the particular details of how specific models of cognitive phenomena built using DSHM are discussed in the relevant papers in this portfolio. The role of this paper is to provide the big-picture, so to speak.

In explaining DSHM, references to ACT-R (Anderson & Lebiere, 1998) will be made, where relevant similarities between DSHM and ACT-R exist. This is done for two reasons. The first is that the reader probably has some knowledge or interest in cognitive modeling and might therefore be familiar with ACT-R. As such, comparisons with ACT-R may help with understanding DSHM. The second reason is that the theoretical

component of this dissertation is to make that case that there is a niche for DSHM that is not filled by ACT-R (or other cognitive architectures). Therefore, illustrating the differences between ACT-R and DSHM will help support this claim.

It is recommended that the reader of this paper read *Dynamically Structured Holographic Memory: An Introduction to the Dissertation* and *Dynamically Structured Holographic Memory: The Mechanisms* prior to reading this document. The numerous references to the latter make this recommendation automatic. The former ought to provide enough of an overview of DSHM to make the latter accessible.

#### *The Origins of DSHM*

The inspiration to create Dynamically Structured Holographic Memory (DSHM) was the result of attending a job talk by Michael Jones at Carleton University in the February 2006. As part of this talk, Jones presented the BEAGLE (Bound Encoding of the Aggregate Language Environment) model of the lexicon (Jones & Mewhort, 2007). BEAGLE was designed as a model of language. However, the general principles that underlie it could be adapted to make it a system for modeling any sort of knowledge.

BEAGLE makes use of a hybrid of discrete and distributed representation. BEAGLE takes sentences consisting of sequences of discrete words as input and outputs words and numerical values. However, internally, words are represented by pairs of vectors of real numbers. One vector, called the environmental vector is a static internal representation of the orthographic form of the word; the other, the memory vector, is dynamic and stores associations between the word and other words in the lexicon. See *Dynamically Structured Holographic Memory: The Mechanisms* for a more detailed

overview of BEAGLE, from a mechanical perspective. For a complete BEAGLE reference, see Jones and Mewhort (2007).

BEAGLE extracts syntactic information about the lexicon from sentences supplied to it. When a sentence is given as input, BEAGLE stores information about the co-occurrence of the words in the sentence, in the memory vectors of the words in the sentence. BEAGLE makes no other separate, explicit record that the sentence has been read. However, the fact that the sentence was read can be recovered from the information stored in the memory vectors of the words appearing in the sentence. Additionally, given a large enough corpus, the memory vectors of the words in the lexicon can be analysed revealing semantic and syntactic features of the language of the corpus.

The memory vectors of the words in the lexicon define a hyper-sphere with 2048 dimensions. The distance between the memory vectors of two words is inversely proportional to their similarity. As such, clusters of words are broadly divided according to their syntactic categories, and more finely organised according to their semantic features.

#### *Differences from BEAGLE*

The most basic difference between BEAGLE and DSHM, from a functional perspective, is that BEAGLE operates only on ordered sequences of words (i.e., sentences), while DSHM can operate on a greater diversity of information objects. The information objects, called items, that DSHM operates on, are collections of other items, and are either ordered (e.g., lists or sequences) or unordered (e.g., sets). As such, DSHM

items can have hierarchical structure (i.e., they can represent tree-like structures), due to the recursive compositionality of items.

This flexibility in information representation available to DSHM allows for a greater variety of memory phenomena to be modelled using DSHM than is possible with BEAGLE. For example, DSHM could consume information objects that represent sentences with hierarchical grammatical structure, or objects that represent visual information such as the relative positions of objects in a scene.

Another difference between BEAGLE and DSHM is in how numerical values are interpreted. In both systems, numerical values are generated by computing the cosines of vectors. There are three sources of vectors: the environmental vectors of items, the memory vectors of items, and probe vectors that are computed based on other vectors. See *Dynamically Structured Holographic Memory: The Mechanisms* for the details of how probe vectors are constructed. In each case, the cosine calculates the similarity of the given vectors. See the *Comparing Vectors: Computing Association Strengths* section below for a description of the cognitive interpretations of various vector comparisons.

For BEAGLE, cosine values are used for two main purposes: ranking the words produced by BEAGLE for completing sentences (a probe vector, environmental vector comparison), and quantifying the semantic similarity of words (a memory vector, memory vector comparison).

For DSHM cosine calculations are also used for ranking completions and quantifying the similarity of items. However, these values are also used as the basis for quantifying other psychological phenomena such as the strength of queued recall, the activation of items used in modeling recognition, and ultimately in calculating retrieval

reaction times. Thus, although DSHM can provide accounts of a greater diversity of memory phenomena than can BEAGLE, due to a re-engineering of the internal mechanics of the system, it relies on the same basic mathematical mechanisms as BEAGLE.

### *Dynamism in DSHM*

In DSHM, each item is a discrete unit, like a chunk in ACT-R DM. However, each item is represented internally by two vectors of real valued numbers. These vectors can be interpreted as patterns of activation in a group of neurons. Each item has a unique signature, its environmental vector, as well as a memory vector that stores the item's associations with other items. See *Dynamically Structured Holographic Memory: The Mechanisms* for the details of how these vectors are computed and manipulated.

This internal, vector based, system of representation in DSHM connects it to dynamical systems theories of cognition (van Gelder, 1998; van Gelder & Port, 1995). Once a DSHM model has been trained on a sufficiently large knowledge base, the vectors representing the items in the system form two large, multidimensional spaces representing aspects of the system. Each vector is a point on the surface of a hypersphere, with a radius of 1.0 and a number of dimensions equal to the number of elements in each vector. The environmental vectors form a state space that defines phases of memory retrieval. The memory vectors define a space that represents the knowledge of the system. These spaces will be referred to as the retrieval space and knowledge space, respectively.

Item completion can be interpreted as a process whereby the state of the system in the retrieval space moves from one representing the initiation of a memory request, to a state that represents the retrieval of an item from memory.

Probe vectors, computed based on the environmental and memory vectors of the context items in the incomplete item, are points in the retrieval space, which represent the states of the system at the initiation of a memory retrieval requests. The attractors in the system are the set of item environmental vectors. As such, the system will gravitate from its initial state to the state defined by the nearest environmental vector. Once in this state, the item corresponding to the environmental vector is retrieved from memory.

The strength of the attraction between the probe and the item is determined by computing the cosine of the probe and environmental vector of the item. The inverse of this cosine value is used to compute the time that elapses between the system's initialization at the position of the probe and eventually reaching the state defined by the position of the environmental vector. This retrieval time is used to determine experimental reaction times.

The environmental vector attractors are fixed, but the memory vectors, from which probe vectors are computed, change as new associations are formed between items, and old associations are reinforced. Thus, the retrieval space is stable, whereas the knowledge space is fluid.

Each location in the knowledge space represents a pattern of associations with every item, collections of items, and possible computations on sets of items, in the system. The space is graded and continuous. When a DSHM model is created, the initial value of the memory vector of each item is equal to the corresponding environmental

vector for that item. As the model learns associations between items the memory vectors in the system change. If the information upon which the model is being trained consists of relatively consistent recurring patterns, the vectors will change rapidly at first, but then settle on stable values that represent the items past histories of association. Therefore, if two items occupy the exact same location in the knowledge space, they have exactly the same roles in the system, (e.g., perfect synonymy in language). However, in practice, similar items are rarely entirely redundant, and subtle differences in their memory vectors reflect subtle differences in their roles in the system. For example, given a diverse enough English language corpus, the near synonyms ‘big’ and ‘large’ would develop similar memory vectors due to the frequency of context in which they are interchangeable and more-or-less equally used. However, the fact that they are not interchangeable in all contexts, such as idiomatic phrases (e.g., ‘big sister’ does not mean ‘large sister’), their memory vectors will differ somewhat.

The distance between two items in the knowledge space is computed by calculating the cosine of two items’ memory vectors. Finding items that share similar meanings is the basis of the ‘cluster method’ for movie recommendation described in Rutledge-Taylor, Vellino and West (2010), included in this portfolio. This ability to perform comparisons between the memory vectors of items allows for the analysis of the relationships between items in a DSHM model.

### *The Atomic Components of an Architecture*

All computational systems are composed to some finite set of parts. In the case of cognitive architectures, these parts represent the finest degree of detail relevant to

providing adequate accounts of cognitive processes. In the case of traditional neural network modeling systems, the node is the most basic part of a model. For most cognitive architectures, the smallest unit of analysis is the production and/or the chunk. DSHM strikes a balance between these two paradigms, and gains the advantages of both.

*The Atomic Components of ACT-R*

One of the core theoretical commitments that ACT-R makes is that the atomic components of thought are declarative memory chunks and procedural memory productions. Chunks represent units of knowledge that are approximately the size of what one can be aware that one knows, e.g., that three plus four equals seven. Each chunk is a small collection of slot value pairs such as that presented in figure 10.

```
Fact3+4:
  isa addition-fact
  addend1 three
  addend2 four
  sum seven
```

*Figure 10.* An ACT-R chunk representing the fact that  $3 + 4 = 7$ .

The chunk illustrated in figure 10 has four slots: isa, addend1, addend2 and sum; which have values addition-fact, three, four, and seven, respectively.

Productions define condition/action pairs that take an if/then structure. The production represents the smallest unit of processing of information in the mind. The *if*

part is sometime referred to as the left-hand-side (LHS) of the production. The *then* part is sometime referred to as the right-hand-side (RHS). Both sides of a production can refer to the contents of one or more buffers. A buffer is a place holder for the currently active chunk associated with a module. The standard modules in ACT-R 6 are the goal module, declarative memory, imagination, vision and motor modules (Bothell, n.d.). On the LHS, the chunks that must be currently held in various buffers for the production to fire can be specified. The values of slots in these buffers can hard-coded or contain variables. On the RHS, new values to be placed in buffers, and retrievals from modules, can be specified.

Two example productions are presented in figures 11 and 12.

```
(P get-fact
  =goal>
    isa remember
    arg1 =a1
    arg2 =a2

==>
  =goal>
    isa write
  +retrieval>
    isa addition-fact
    addend1 =a1
    addend2 =a2
)
```

*Figure 11.* An ACT-R production for remembering an addition fact.

```

(P write-fact
  =goal>
    isa write
  =retrieval>
    isa addition-fact
    sum =thesum
==>
  =goal>
    isa remember
    arg1 =thesum
!output! =thesum
)

```

Figure 12. An ACT-R production for writing an addition fact sum.

The get-fact production fires if the LHS is satisfied; i.e., the chunk in the goal buffer is of the type remember. The *a1* and *a2* variables are bound to whatever values are in the goal chunk. The RHS specifies that the goal should be changed to a chunk of type write and that an addition-fact chunk from memory, that matches the *addend1* and *addend2* slots, should be retrieved.

The write-fact production fires if the goal chunk is of the write type, and an addition-fact chunk has been retrieved from DM. The production changes the goal chunk

back to a remember chunk, but with the first argument set to the retrieved sum. The production also writes the retrieved sum to the system output in a human readable form.

The write-fact production will fire as long as there are chunks in DM that match the retrieval condition of the get-fact production.

### *The Atomic Components of DSHM*

The atomic components of DSHM are items, which correspond to ACT-R chunks and slot values. ACT-R chunks correspond to complex items and ACT-R values correspond to DSHM atomic items.

The simplest formulation of DSHM item that represents the fact represented by the chunk in figure 10 is presented in figure 13.

```
[isa:addition-fact addend1:three addend2:four sum:seven]
```

*Figure 13.* A DSHM item representing the fact that  $3 + 4 = 7$ .

In order to prompt a DSHM model to retrieve an addition fact where the first addend is three and the second addend is four, an incomplete item (with the item representing the sum replaced with a query item) is presented to the model; see figure 14.

When instructed to complete the supplied incomplete item in figure 14, the DSHM model would return a list of completions of the item. The model can be instructed to return only what it deems to be the top completions of the incomplete item. This is interpreted as the model recalling a set of plausible completions

```
[isa:addition-fact addend1:three addend2:four sum:?thesum]
```

*Figure 14.* An incomplete item used for prompting the recall of an addition fact with one addend specified.

In a DSHM model, the work that would be done by productions in ACT-R is written into the program code that implements the DSHM model. For example, the list of completions could be parsed and the top completions extracted, and fed into another function call. The fact that retrieval in DSHM is a single step process helps mitigate the need for an elaborate production system.

Should DSHM evolve into a full-fledged cognitive architecture a production system, of sorts, would be added. One possibility is to remain consistent with the ad hoc nature of items, and make productions ad hoc as well. For example, the left and right hand sides would be given vector representations, and would be associated together. Then given a LHS representation, filled with particular item values, possible RHS would be ranked, with the top-most (or a probabilistically determined) RHS being selected and executed.

However, at present, DSHM accounts only for the associative relationships between items. As such, the atomic components of DSHM consist only of items.

### *Small Atoms*

An important difference between DSHM and ACT-R is that DSHM is not committed to items representing facts that one can be aware that one knows. Rather, items should represent the smallest units of interest to the modeller. For example, if

adopting the granularity of ACT-R DM, the phrase “the red car”, would be represented as the item in figure 15. In this case, the phrase is represented by a complex item with three ordered sub-items, each of which is a word. Each word is an atomic item. Here the atoms of cognition are words.

[the:red:car]

*Figure 15.* A representation of “the red car”, where words are the atoms of cognition.

However, if the purpose the model was to help understand word pronunciation, the item in figure 16 might be more appropriate representation of the phrase. In this case, the phrase is represented by a complex item with three ordered sub-items, each of which is a word. Each word is a complex item with ordered sub-items, each of which is an atomic item representing a phoneme. The level of detail presented in figure 16, where phonemes are the atoms, would be appropriate for models of phoneme restoration (Samuel, 1981).

[[ð:ə]:[r:ɛ:d]:[k:a:r]]

*Figure 16.* A representation of “the red car”, where phonemes are the atoms of cognition.

If a finer level of detail is needed, the phrase could be represented by the item in figure 17. In this case, the phrase is represented by a complex item with three ordered sub-items, each of which is a word. Each word is a complex item with ordered sub-

items, each of which is a phoneme. Each phoneme is a complex item with unordered sub-items, each of which is a phonetic feature. In this case, it should be noted that each item has no more than Miller's seven, plus or minus two, sub-items (Miller, 1956). However, a great deal of information is packed into the root item due to the hierarchical structure that accommodates complex components.

```

[[[voiced dental fricative]:[central close-
mid]]:[[alveolar trill]:[open-mid near-front]:[voiced
alveolar plosive]]:[[unvoiced velar plosive]:[open
central]:[alveolar trill]]]

```

*Figure 17.* A representation of “the red car”, where phonetic features are the atoms of cognition.

Deeply structured items such as that presented in figure 17 are meant to account for how it is that people are able to comprehend rich information sources without becoming overwhelmed. Here each phoneme consists of only a handful of features, each word consists of only a few phonemes, and each phrase consists of only a few words.

#### *Ad Hoc Items*

The most significant difference between ACT-R and DSHM pertains to the use of recurring patterns of knowledge. ACT-R makes use of chunk types, while all complex items in DSHM are ad hoc. The definition of an ACT-R chunk type consists in a set of

slots. This dictates that chunks with this exact set of slots are permissible. All chunks in an ACT-R model must be instances of a chunk type.

In contrast, complex items, in DSHM, are defined, on the fly, in code. This is possible because the degree of association between any two items can be determined regardless of whether they have been explicitly associated in the past history of the model, or not.

Any arbitrary combination of sub-items can be assembled to make a complex item. The complex item can be judged to be familiar, or not, based on how strongly associated the sub-items are to one another. If the exact complex item has been presented to a DSHM model previously, then the item will likely resonate strongly and the model will judge it to have been perceived in the past. If on the other hand, none of the sub-items bear any relationship to one another, the item will not resonate strongly and the model would judge it to be a novel construction.

Interestingly, if some of the parts of the item in question have co-occurred before, as part of other items, the given item may resonate enough cause the model to judge that the item is familiar. This may cause the model to make an error in the case of foils in a memory recall task (which is a good thing, because humans make recall errors). It is also the key feature that allows DSHM the flexibility to make use of knowledge when it is primed for recall in a slightly different form than when it was learned. See the *Geography Fact Conversion* example in the appendix, for an illustration of this.

#### *Complex Items in Memory*

The fact that items are ad hoc in DSHM does not mean that there are no complex items stored in DSHM. Any complex item that is the sub-item of another complex item

is stored as a cohesive unit in DSHM. For example, if the item shown in figure 16 is presented to a DSHM model, 11 additions are made to the set of items in the model's memory (if they did not already exist). They are the atomic items  $\delta$ ,  $\epsilon$ ,  $r$ ,  $\epsilon$ ,  $d$ ,  $k$ ,  $a$ , and  $r$ , and the complex items  $[\delta:\epsilon]$ ,  $[r:\epsilon:d]$ , and  $[k:a:r]$ . The item  $[[\delta:\epsilon]:[r:\epsilon:d]:[k:a:r]]$  is not added, however. This is because associating its sub-items together is both necessary and sufficient for storing it in memory.

Complex items are added to memory so that they can serve as substitutes for query items. See *Dynamically Structured Holographic Memory: The Mechanisms*, for a complete account of the mechanisms underlying memory retrieval. This makes a correct completion of  $[[\delta:\epsilon]:[r:\epsilon:d]:?x]$  possible where the structure of the missing item is unknown (i.e., it is not necessarily an atomic item). If the structure of the missing item is known in advance, substituting complex items for query items is not necessary. For example, the completion request could be formatted  $[[\delta:\epsilon]:[r:\epsilon:d]:[?x:?y:?z]]$ , if it is known that only ordered complex items with three sub-items are possible candidates for satisfying the third sub-item of the root incomplete item. In the case of  $[[\delta:\epsilon]:[r:\epsilon:d]:?x]$ ,  $?x$  would be satisfied with  $[k:a:r]$ ; in the case of  $[[\delta:\epsilon]:[r:\epsilon:d]:[?x:?y:?z]]$ ,  $?x$  would be satisfied with 'k',  $?y$  with 'a', and  $?z$  with 'r'.

The important difference between DSHM and ACT-R in the use of stored complex items (chunks) is that a stored complex item plays no role in the retrieval of that item. For example, if a DSHM model is instructed to complete the item  $[k:a:?x]$ , only the atomic items 'k' and 'a' serve as the basis for determining what items should be made candidates for completing the incomplete item. Thus, according to this hypothetical model, the word represented by  $[k:a:r]$ , is a known word, not because it exists as an

explicit entry in a mental lexicon. Rather it is a known word because the phonemes ‘k’, ‘a’, and ‘r’ are strongly associated with one another.

### *Association*

Association in DSHM is consistent with the independent associations hypothesis (IAH), as defined by Rizzuto and Kahana (2001). That is, association is a unidirectional relationship. However, when an association is formed between two items, two separate unidirectional processes take place, making the result indistinguishable from the effect of one bidirectional process. In practice, this distinction is irrelevant during the formation of associations. However, it is very relevant when it comes to computing the strengths of association between items. This is because the strengths of association between two items are not necessarily symmetrical. That is, A can be more closely associated with B, than is B to A. This is because there is a constant amount of association that an item bears to all other items. The capacity for association is determined by the length of the vectors used to represent items’ memory vectors (See *Dynamically Structured Holographic Memory: The Mechanisms*). Thus, if A is associated only with B, the strength of the association between A and B will be quite strong. However, if B is associated with A, C, D, E and F, the strength of the association between B and A will be relatively weak. DSHM can make retrieval errors when the memory capacity of context items (i.e., primes) are saturated and produce noisy association information.

As a note to the reader, it should be kept in mind that when it is written that the strength of association between two items is computed, this should *not* be taken to mean

that a single value representing a bidirectional strength is the result; the result is two values, one for each separate association strength (e.g., A to B, and B to A).

### *Comparing Vectors: Computing Association Strengths*

As discussed in the *Dynamism in DSHM* section above, the cosine of two items' memory vectors is measure of the semantic similarity of the items. From a psychological perspective, this is the process of focussing on one idea or concept and searching (either consciously, or not) for other ideas that share common features. Similar ideas will be recalled first, because they share common meaning, from a psychological perspective, while from a mechanical perspective it is because the items representing them are located near to one another in the knowledge space.

It is worth noting that the vectors used to represent items in DSHM are not feature lists. Each element, individually, does not possess any semantic content. Rather, an entire vector has content based on its location in the knowledge space (discussed in the *Dynamism in DSHM* section above).

The cosine of one item's environmental vector and another item's memory vector is a measure of how often the first item co-occurred, as sub-items of an unordered complex item, with the second item, relative to how often other items had co-occurred with the second item. This value will be correlated with the cosine of the second item's environmental vector and the memory vector of the first item, under most circumstances. The values are not necessarily equal since one item can have a different total number of associations than the other item.

In this case, the items do not necessarily share any common semantic features, but may due to the fact that unordered sets of items tend to be of the same category. For example, pairs or phonetic features (as in those in the item shown in figure 17) will never become synonymous, by definition. However, they will occupy the same general area of the knowledge space because they tend to be associated with the same sets of neighbours.

To determine whether one item has occurred to the left (or right) of another item, in the set of sub-items of an ordered complex item, a more complex algorithm must be applied. See *Dynamically Structured Holographic Memory: The Mechanisms*. If the DSHM model is operating on items with structures that adhere to formal grammars (Linz, 1997), the neighbouring items may be of different grammatical categories, and therefore be unlikely to share common features or occupy common regions of the knowledge space.

Testing for items that have co-occurred in an ordered complex item is the basis for recalling sequences. For example, if someone is trying to remember the correct musical note in a song, they can sing (or, play) a portion of the song preceding the note they are trying to remember, as a recall strategy. The known sequence of notes primes the recall of the desired note. In DSHM, this is a matter of computing which note (i.e., the item representing the note) is the most likely to follow the notes provided as context. A large context provides the model with more information and allows it to predict a note with more confidence.

### *Learning*

In DSHM there is only one basic mechanism for learning. This is the formation of associations between items in memory. Items are associated by presenting them to a DSHM model, as sub-items of a complex item. For example, the words *the*, *red*, and *car*, could be associated in a model by presenting the complex item [the:red:car] to the model. Each item would be associated with each other item, and if the appropriate parameter is set (see *Dynamically Structured Holographic Memory: The Mechanisms* for a detailed account of the association formation process), the convolution of each pair of items would be associated with the other item as well. In this case, the item [the:red:car] would not be associated with any other item. However, if the complex item [[the:red:car] [goes:fast]] is presented to the model, the complex items [the:red:car] and [goes:fast] would be associated together.

Because items are ad hoc, presenting [the:red:car] to DSHM, would increase the resonance of [my:red:car] due to the reinforced association between *red* and *car*.

### *Reinforcement*

Unlike ACT-R chunks, all items in DSHM are ad hoc and lack persistence. As such, it is impossible to assign an activation strength to an item, and therefore impossible to reinforce that strength. However, the resonance of an item can be increased by presenting the item to the system. With each exposure, the memory vectors of the items become more closely associated with the environmental vectors of the other items, resulting in higher resonance scores. Thus, reinforcement is a feature of DSHM.

### *Activation*

Items in DSHM have no base level activation in the same sense that chunks in ACT-R DM do. In DM, the base-level activation of a chunk is a context independent estimate of how likely the chunk is to match a given production. This is based on how recently and frequently the chunk has been used (Anderson & Lebiere, 1998).

The equation for calculating base level activation of a chunk,  $C_i$ , in ACT-R, is shown in figure 18.

*Figure 18.* A formula for calculating the base level activation ( $B_i$ ) of a chunk  $C_i$ , in ACT-R, where  $n$  is the number of presentations of  $C_i$ ,  $t_j$  is the time since the  $j$ th presentation of  $C_i$ , and  $d$  is the decay parameter, which is usually set to 0.5.

Where a chunk,  $C_i$ , has occurred only once, the equation simplifies to:  $B_i = \beta - d \ln(t)$ , where  $t$  is the time since  $C_i$  occurred. Since,  $\beta$  is often set to 0 by default:  $B_i = -d \ln(t)$ , which is a close approximation to the natural logarithm of the DSHM cosine formula, presented in figure 4, which is used to calculate confidence values; see figure 19.

$$\ln(1 / (x + 2)^{1/2}) = -0.5 \ln(x + 2) \approx -0.5 \ln(x)$$

*Figure 19.* The natural logarithm of the canonical cosine formula in DSHM.

The cosine formula described above is used to calculate the degree of association the item B bears to the item A, where the items have been associated together once, and  $x$  subsequent associations have been made between A and other items. The variable  $x$  can also represent the ratio of the total number of associations the item A has made, and the total number of times B has been associated with A. DSHM is atemporal and therefore, the times at which the associations were made is not a factor in calculation of association strength.

The significance of this relationship between the formulae for calculating base-level association in DM and confidence values in DSHM is that these confidence values are used in calculating the resonance of a complex item in DSHM.

Given a complex item, the resonance of the item can be calculated. This is a measure of how tightly associated the sub-items of the complex item are. Thus, there is a sort of equivalence between base-level activation in DM and resonance in DSHM. However, this similarity should not obscure the fact that the meanings of base-level activation and resonance are different, and depend on different quantifiable aspects of the given chunks and items.

In DSHM, an item's resonance increases when the item is presented to the model a subsequent time (i.e., it is reinforced), and its resonance decreases when any of the item's sub-items form new associations with other items that are not sub-items of the given item. Therefore, the resonance of an item decays due to interference only. See the *Memory Degradation* section of this paper for more details on how DSHM accounts for decay.

It is worth emphasising here that an item can score a high resonance without ever having existed previously. This could happen if various combinations of the sub-items in the item have co-occurred together before, and with sufficient frequency.

### *Spreading Activation*

DSHM can be interpreted as possessing a form of spreading activation similar to that of ACT-R. In ACT-R, the activation function includes the term show in figure 20.

*Figure 20. Spreading activation, in ACT-R.*

In ACT-R the amount of activation a chunk receives from other chunks is computed using this formula. For every value in every chunk of every buffer, every chunk in DM that shares one of those values gets a boost in its activation. The amount of this boost is inversely proportional to the fan of the value ( $W_j$ ), and is scaled by the strength of the association between the chunks ( $S_{ji}$ ).

For example, given the 'get-fact' production shown in figure 11, the values of the arg1 and arg2 slots in the goal buffer chunk, will boost the activation of addition-fact chunks in DM with these values.

In DSHM, retrieval probes can be thought of as the source of activation which spreads throughout the retrieval space (discussed in the *Dynamism in DSHM* section above). The activation of items in the retrieval space is inversely proportional to their distances from the probe.

### *Partial Matching*

Partial matching does not exist in DSHM. The items retrieved from memory must include each of the context items (i.e., primes) provided. For example, if a DSHM model is instructed to complete the incomplete item in figure 14 (above), the item in figure 13 would be a possible completion, while the item shown in figure 21 would not. This is because the sub-item [addend2:five], was not part of the original incomplete item presented to the model. A partial match based on the other context items [isa:addition-fact] and [addend1:three] is not possible.

```
[isa:addition-fact addend1:three addend2:five sum:eight]
```

*Figure 21.* A DSHM item representing the fact that  $3 + 5 = 8$ .

However, the ad hoc nature of items in DSHM allows for similar retrieval consequences as does partial matching in ACT-R. For example, if an ACT-R model is attempting to retrieve the answer to the question, “what is three plus four”, it can solve this problem correctly by retrieving the chunk that represents, “three plus four is seven”. A common way for ACT-R to account for errors is to allow for partial matching. If the ACT-R model were to retrieve the chunk representing, “three plus five is eight”, it could incorrectly report that the answer to the question, “what is three plus four” is ‘eight’. A DSHM model of addition could make the same error by ranking the incorrect completion shown in figure 22 ahead of the correct completions shown in figure 13.

```
[isa:addition-fact addend1:three addend2:four sum:eight]
```

*Figure 22.* A DSHM item representing the incorrect proposition that  $3 + 4 = 8$ .

The item shown in figure 22 is a possible retrieval because it is an ad hoc construction based on interfering associations. It could out rank the correct completion, if the strengths of the associations between [sum:eight] and the other sub-items were stronger than those for [sum:seven]. Normally, the influence of the memory queue based on the convolved tuple (addend1:three addend2:four), which is associated with [sum:seven] but not [sum:eight], is sufficient to cause the retrieval of the correct math fact. However, if the uLambda parameter is set to 1, this tuple would not be computed (because it has more than one element). Additionally, if the items representing the facts “three plus five equals eight” and “four plus four equal eight” have been reinforced much more than the fact “three plus four equals seven”, the strong associations between the addends ‘three’, the addend ‘four’, and the [isa:additon-fact] item, and the sum ‘eight’, could, on their own, boost the incorrect answer to the point that it out ranked the correct answer.

It is worth pointing out that this hypothetical retrieval error does not require noise. This incorrect retrieval is the result of one set of correct facts interfering with another correct fact.

### *Noise*

DSHM is deterministic. As such, it embodies no stochasticity, per se. However, a DSHM model assigned a small vector length will make more errors than a DSHM model

assigned a larger vector length due to memory saturation (Rutledge-Taylor & West, 2008); see figure 26. Additionally, it is not possible, in practice, to predict which errors will occur. Thus, in DSHM noise is emergent upon memory saturation and is parameterized by vector length. As such, it is a global parameter for each model.

### *Memory Degradation*

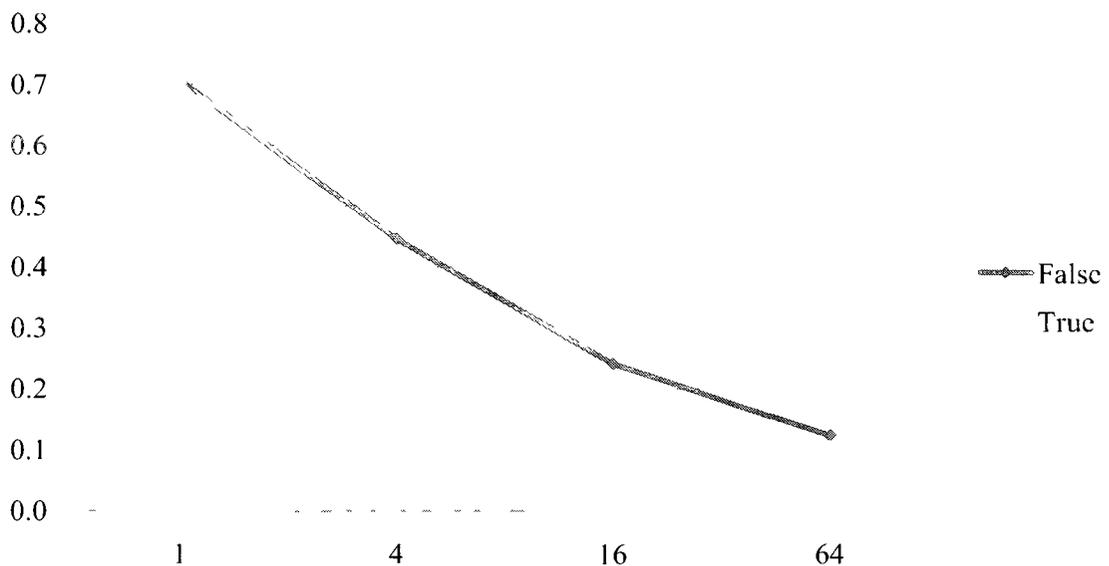
DSHM does not include any explicit mechanisms devoted to supporting association strength decay over time. According to DSHM theory, decay is the result of interference only. On average, resonance values across time approximate a decreasing exponential function, where reinforcement is absent, because interference is ubiquitous and continuous. DSHM is meant to model systems where large amounts of information, that touch a wide range of items in memory, is being reconciled and accommodated continuously.

DSHM was originally designed to model long-term memory only. As such, an account of the rapid decay characteristic of new memories was not originally included in DSHM. However, a global decay parameter, which is set to false by default, was later added. When set to true, associations decrease more with each new association. Setting *decay* to true does not imbue DSHM with any sensitivity to time, however. See the *Defining an Instance of DSHM* section of *Dynamically Structured Holographic Memory: The Mechanisms* for the details of how this parameter works.

Figure 23 shows the decrease in the association of an item B has to an item A over time, as item A forms additional interfering associations (not with B), after an initial association with B. The trends for both values of the decay parameter are shown.

### *Reaction Times*

There is no official formula relating confidence values, or resonance values to reaction time. This is because, DSHM does not account for cognitive processes that lie outside of pure recall and recognition. However, scaling the inverse of confidence (or resonance) values by a constant has consistently proven to provide good fits to human reaction time data (Rutledge-Taylor, Pyke, West & Lang, 2010; Rutledge-Taylor & West, 2008).



*Figure 23.* The association strengths between two items as additional interfering associations are made. The  $x$  axis represents the number of interfering associations. The  $y$  axis represents association strength. One line depicts the trend when the decay parameter is set to true; the other when it is set to false.

A short term goal in developing DSHM is to review the mechanisms for producing confidence values and formalize an official formula for converting these values to recall and recognition times.

*Ecological Validity: Models Follows the Same Procedures as Human Participants*

A point to be made which is not central to the theory of memory retrieval in DSHM is the value placed on ecological validity. All of the DSHM models presented in this portfolio were trained in a manner that resembles as closely as possible, the regime under which human experimental participants were trained. For example, in Rutledge-Taylor and West (2008), great effort was made to ensure that the DSHM model proceeded through the phases of the experiment (study, recall, and recognition) exactly as did the human participants. In the recall phase, the model was required to repeat the recall phase until it was able to recall all and only the correct answers to all of the recall questions.

Training a model in the manner in which human participants are trained ensures that no incorrect simplifying assumptions are made about the learning and reporting processes, which could adversely affect the model's ability to match human performance.

Neural Plausibility

The claim is made that DSHM connects to neurally inspired models of memory. This is not the same as claiming that DSHM is neurally plausible. However, all neurally inspired modeling systems ought to have something to say about how they relate to the biological neuronal systems. In the case of artificial neural networks, the relationship is often not pressed any further than to draw an analogy between the roles of network nodes and neurons, where connections represent synapses, etc (e.g., Rosenblatt, 1958;

Rumelhart, Hinton & Williams, 1986). The notable exception is the work done by computational neuroscientists, who explicitly model the precise details of information processing in biological neurons (e.g., Churchland & Sejnowski, 1992; Eliasmith & Anderson, 2003; Stewart, Choo & Eliasmith, 2010; Stewart & Eliasmith, 2008).

In the case of DSHM, there are no nodes that play the role of neuronal units. Rather, information is represented in a set of vectors. Here the analogy is between vectors and patterns of activation in populations of neurons. Information is stored and extracted by adding and comparing vectors together. Simple associations are based on simple addition, while complex relationships are encoded using convolution, and decoded using correlation. It has been argued that convolution and correlation are not neurally plausible memory processes (Pike, 1984). However, the counter-argument has been made that correlation is a common feature of information processing in the human perceptual system, and therefore is a plausible memory process, although not a proven memory process (Murdock, 1985).

### *The Holonomic Theory of Mind*

There is no established direct relationship between the holography as it exists in DSHM, and the holonomic theory of mind (Pribrum, 1987; 1991). In the case of DSHM, each entity in the mind, be it the perception of an object, or some other sort of concept, is represented by an item, which stores the associations between that item and other items holographically. In the case of the holonomic theory of mind, it is the patterns of activation in the dendritic arbor of neurons which possess holographic properties. Thus, the scale at which the holography applies is different in the two theories. However, in its

present form DSHM does not make any commitments to its biological implementation. Therefore, the potential exists for the holography in DSHM to connect to holonomic theory of mind at more than an abstract level, in the future. Nevertheless, the fact that holonomy plays a central role in information representation in both theories supports the idea that holographic representation could be a pervasive feature of cognition.

## Appendix

### *Geography Fact Conversion Example*

In ACT-R, in order to convert information contained in chunks of one set of chunk types to a chunk of a different chunk type, chunk specific conversion productions need to exist. Thus, the conversion process is serial and rule-governed. In DSHM, the use of ad hoc items is automatic because there are no persistent item types. According to DSHM theory, the recall of ad hoc items is a distributed, primitive memory process.

For example, let us consider a hypothetical ACT-R model that specifies the following chunk types: (chunk-type geography-fact place borders direction), and (chunk-type demographic-fact place speaks percentage).

Example chunks are: (isa geography-fact place quebec borders new\_york direction south), and (isa demographics-fact place quebec speaks french percentage:95)

In order to combine this information so as to cause the model to recall a place that borders New York and where the populace speaks French, several carefully designed productions that retrieve the relevant chunks and combine the information would need to be created. To store this new fact, for later recall, a new chunk type would be needed as well: (chunk-type combo-fact place borders speaks).

A problem with this is explored in Rutledge-Taylor (2005). The issue is that all the permutations of combinations of chunk slots that could possibly be required by the model would need to be anticipated in advance of the model running. For each combination, sets of productions for extracting the needed information from DM would need to be created. Additionally, if a set of values are to be pushed back to DM, a chunk type for that combination of values would need to be created in advance as well.

This is not a problem for a model of a single task, where information is applied in the same way that it is consumed, and therefore does not need to be altered in any way. However, it is a potentially a huge problem for a model that could require any combination of chunk values be combined so as to be relevant for engaging in some arbitrary task or other.

#### *The DSHM Solution*

DSHM avoids the problem of context specificity in items because it does not make use of any persistent chunk types; *all items are ad hoc*. Associations between items are always formed in one context or other. However, the knowledge stored in the memory vector of an item is a composite of all the contexts in which it has occurred. The idiosyncratic details of the specific contexts interfere with one another and are eventually lost, and what remains is a context-free core of the ubiquitous properties of the item.

For example, the item representing 'chair' does not explicitly include a finite set of features that define it. Rather, it embodies many different strong associations with other items, such as 'legs', 'back', 'seat', 'sitting', 'folding', 'office', 'department', 'chairman', 'orchestra', 'railway', etc, but no persistent strong associations with

particular chairmen or offices. However, given enough context, ‘chair’ can contribute the recall of a particular chairman.

Given any arbitrary context, the relevance of a particular item, such as ‘chair’, can be determined by calculating the degree to which it is associated with the items in the given context. The sensitivity of items to combinations of co-occurring items allows items to bear non-linear relationships to other items. For example, ‘chair’ might bear a negligible association with each of the items, ‘rail’, ‘steel’, and ‘sleeper’, individually, but might be strongly associated with all three when presented together, as in *chair*, the brace on a railway tile for holding a rail.

A DSHM model designed to process the same information given to the ACT-R model discussed above, could be presented with the complex items: [isa:geography-fact place:quebec borders:new\_york direction:south], and [isa:demographics-fact place:quebec speaks:french percentage:95].

For the purposes of this example, these items convey the same information as the chunks listed above. To have the model recall a place that borders New York and whose populace speaks French, it would complete [?x borders:new\_york speaks:french].

The result [place:quebec borders:new\_york speaks:french], where [place:quebec] is substituted for ?x, is produced in a single (externally represented) step. Although, other items might be strongly associated with [borders:new\_york] and [speaks:French], [place:quebec] has the greatest combined associations to the provided cues; see figure 24. It should be noted that, internally, the code that accomplishes this task consists of many steps. Nevertheless, according to DSHM theory, this recall process takes a single step (composed to many parallel processes).

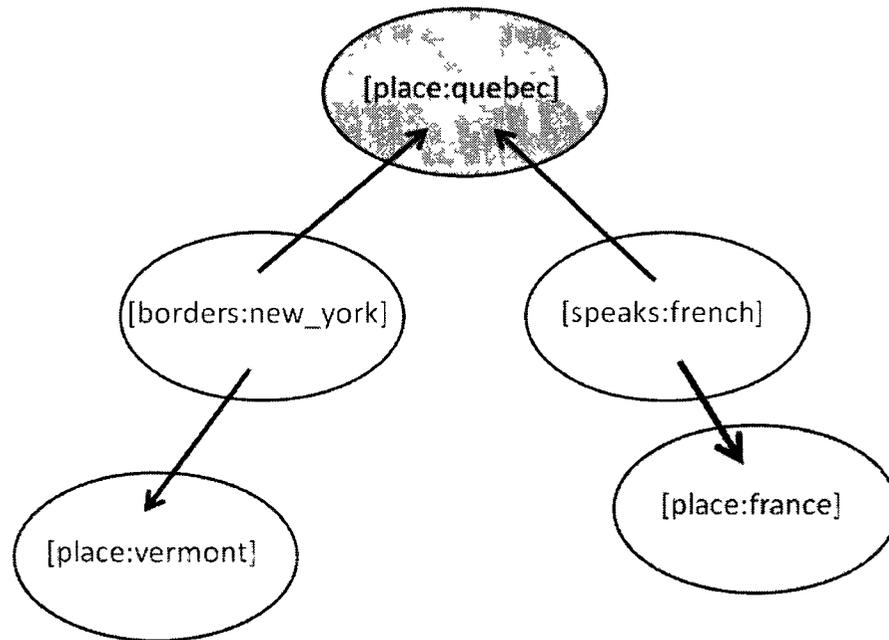


Figure 24. Combined associations to [borders:new\_york] and [speaks:french]; darker shade indicates more association.

The DSHM method of retrieval simplifies the mechanics of creating ad hoc items, relative to ACT-R. This is an important step towards solving knowledge representation problems, discussed in *Dynamically Structured Holographic Memory: An Introduction to the Dissertation* and Rutledge-Taylor (2005).

#### References

- Anderson, J. R., & Lebiere, C. (Eds.) (1998). *The atomic components of thought*. Mahwah, NJ: Lawrence Erlbaum.
- Bothell, D. (n.d.). ACT-R 6.0 Reference Manual. Retrieved August 11, 2010, from <http://act-r.psy.cmu.edu/actr6/reference-manual.pdf>

- Churchland, P. S., & Sejnowski, T. J. (1992). *The computational brain*. Cambridge, MA: The MIT Press.
- Eliasmith, C., & Anderson, C. H. (2003). *Neural engineering: Computation, representation and dynamics in neurobiological systems*. Cambridge, MA: MIT Press.
- Jones, M. N., & Mewhort, D. J. K. (2007). Representing word meaning and Order information in a composite holographic lexicon. *Psychological Review*, *114*, 1-37.
- Linz, P. (1997). *An introduction to formal languages and automata*. Surrey, MA: Jones and Bartlett Publishers.
- Murdock, B. B. Jr. (1985). Convolution and matrix systems: A reply to Pike. *Psychological Review*, *92*. 130-132.
- Pike, R. (1984). Comparison of convolution and matrix distributed memory systems for associative recall and recognition. *Psychological Review*, *91*, 281-294.
- Pribram, K. H. (1987). The implicate brain. In B. J. Hiley and F. D. Peat (Eds.). *Quantum implications: Essays in honour of David Bohm* (pp. 365–371). London: Routledge.
- Pribram, K. H. (1991). *Brain and perception: Holonomy and structure in figural processing*. Hillsdale, NJ: Lawrence Erlbaum.
- Rizzuto, D. S., & Kahana, M. J. (2001). An auto-associative neural network model of paired-associate learning. *Neural Computation*, *13*, 2075-2092.
- Rutledge-Taylor, M. F., Pyke, A. A., West, R. L. & Lang, H. (2010). Modeling a three term fan effect. In D. D. Salvucci & G. Gunzelmann (Eds.), *Proceedings of the*

*Tenth International Conference on Cognitive Modeling* (pp. 211-216). Philadelphia, PA: Drexel University.

Rutledge-Taylor, M. F., Vellino, A. & West, R. L. (2010). Dynamically structured holographic memory for recommendation. Carleton University Cognitive Science Technical Report 2010-01. URL <http://www.carleton.ca/ics/TechReports>

Rutledge-Taylor, M. F. & West, R. L. (2007). MALTA: Enhancing ACT-R with a holographic persistent knowledge store. In D. S. McNamara & J. G. Trafton (Eds.), *Proceedings of the 29<sup>th</sup> Annual Conference of the Cognitive Science Society* (pp. 1433-1438). Nashville, TN: Cognitive Science Society.

Rutledge-Taylor, M. F. & West, R. L. (2008). Modeling The fan effect using dynamically structured holographic memory. In B. C. Love, K. McRae, & V. M. Sloutsky (Eds.), *Proceedings of the 30<sup>th</sup> Annual Conference of the Cognitive Science Society* (pp. 385-390). Washington, DC: Cognitive Science Society.

Samuel, A. G. (1981). Phonemic restoration: Insights from a new methodology. *Journal of Experimental Psychology: General*, 110, 474-494.

Stewart, T. C., Choo, X., & Eliasmith, C. (2010). Dynamic behaviour of a spiking model of action selection in the basal ganglia. In D. D. Salvucci & G. Gunzelmann (Eds.), *Proceedings of the Tenth International Conference on Cognitive Modeling* (pp. 235-240). Philadelphia, PA: Drexel University.

Stewart, T. C. & Eliasmith, C. (2008). Building production systems with realistic spiking neurons. In B. C. Love, K. McRae, & V. M. Sloutsky (Eds.), *Proceedings of the 30<sup>th</sup> Annual Conference of the Cognitive Science Society* (pp. 1759-1764). Washington, DC.

Van Gelder, T. (1998). The dynamical hypothesis in cognitive science. *Behavioral and brain sciences*, 21, 615-628.

Van Gelder, T. J., & Port, R. (1995). It's about time: An overview of the dynamical approach to cognition. In R. Port & T. van Gelder (Eds.), *Mind as motion: Explorations in the dynamics of cognition*. (pp. 1-43). Cambridge, MA: MIT Press.

## CAN ACT-R REALIZE “NEWELL’S DREAM”?

Previously published as Rutledge-Taylor, M. F. (2005). Can ACT-R realize “Newell’s Dream”? In B. G. Bara, L. Barsalou & M. Bucciarelli (Eds.) *Proceedings of the 27<sup>th</sup> Annual Conference of the Cognitive Science Society* (pp.1895-1900). Stresa, Italy: University and Polytechnic of Turin.

### Contribution

Written before DSHM was designed, this paper establishes the basis for the theoretical concerns pertaining to the limitations of ACT-R DM. DSHM was created so as to provide a memory modeling system that might be less susceptible to the concerns described below.

### Abstract

In “The Atomic Components of Thought”, John Anderson and Christian Lebiere claim that ACT-R (4.0) realizes “Newell’s Dream” of a unifying theory of cognition. In this paper it is suggested that each ACT-R model can account for only a finite set of cognitive processes, and cannot therefore be used to model an unbounded whole mind. It is suggested that this is due to an inherent context dependence of ACT-R models. This limitation runs counter to the intuitive criterion that a unifying theory of cognition ought to be able to provide an account of the mind as a system not bound to any particular context. It is suggested that thought in cognitive models ought to be conceived as temporary context specific operations based on persistent context independent knowledge. The basis for a new cognitive architecture, which differentiates thought from

knowledge is proposed. This new architecture combines ACT-R with elements of Lawrence Barsalou's situated simulation theory.

### Introduction

In 1972, at the Carnegie Symposium on Cognition, Allan Newell raised a concern about the course of research in psychology. He delivered a paper entitled “You can’t play 20 questions with nature and win”, in which he lamented the fact that there was very little that unified the wealth of knowledge that had been accumulated about individual human cognitive processes (1973a). In a paper published separately in the proceedings of the symposium, Newell suggested that production systems might serve as detailed models of the human control structure (1973b; 1990). Eighteen years later, Newell published a book entitled “Unified Theories of Cognition” in which he proposed that cognitive architectures hold the key to unifying psychology (1990).

There are many different cognitive architectures used to produce cognitive models of psychological phenomena, including Newell’s own SOAR architecture, which was first released in 1982 (Laird & Rosenbloom, 1996). However, the most popular architecture is ACT-R (Anderson, 1993; Anderson & Lebiere, 1998). This popularity is by no means accidental. The theory of cognition it implements has been well developed, and hence, has allowed a wide variety of researchers to produce theoretically grounded models of various cognitive phenomena. Additionally, and perhaps most significantly, ACT-R models typically fit the human experimental data they are designed to model, quite well.

This paper considers whether ACT-R, as it currently exists, realizes “Newell’s Dream” of a unifying theory of cognition, or not. It is suggested that given appropriately strict criteria ACT-R may be inadequate.

### *Newell’s Criteria*

According to Newell: a theory is an explicit body of knowledge, from which answers to questions of a predictive, explanatory, or prescriptive type can be given; theories are approximate; theories cumulate; and, theories develop iteratively (1990, pp. 13-14). Newell defines a unified theory of cognition as “a single set of mechanisms for all of cognitive behavior” (1990, p. 15). He specifies these mechanisms as a prioritized list of areas of cognitive phenomena to be covered. They are, in order: problem solving, decision making, and routine action; memory, learning, and skill; perception, and motor behaviour; language; motivation, and emotion; and, imagining, dreaming, and, daydreaming. Thus, a complete unified theory of cognition, should account for all of these cognitive phenomena. However, given Newell’s views on theory development, an acceptable strategy would be to begin with a unified theory of the phenomena at the top of the list, and slowly augment the theory so as to accommodate successive items.

### *Background Theory*

Ubiquitous in cognitive science is the view that cognitive systems can be analysed from a variety of perspectives. The tri-level hypothesis is that there are three basic levels of analysis (Dawson, 1998). Various researchers apply their own labels to these three levels. Newell divided them into the biological, cognitive, and rational (Newell, 1990); Zenon Pylyshyn makes use of the physical, syntactic, and semantic (Pylyshyn, 1999);

and, Michael Dawson, the implementational, algorithmic, and computational levels (Dawson, 1998). Despite the difference in terms, there is arguably an equivalence between these hierarchies. The biological, physical, and implementational levels are, in the case of humans, the levels of description that (typically) appeal primarily to neural processes. The cognitive, syntactic, and algorithmic levels, describe human cognition in terms of operations on syntactic (or, otherwise, formal) structures. The rational, semantic, and computational levels are those at which the cognitive system is described in terms of its knowledge (i.e., goals, beliefs, and perceptions etc.).

Pylyshyn asserts that a fundamental hypothesis in cognitive science is that this knowledge level is an autonomous (or, at least, partially autonomous) level of description (1998, p. 4). That is, it is autonomous from the neural level of description. This conclusion depends on another belief that is ubiquitous to cognitive science: the computational theory of mind. The computational theory of mind is that the mind is a kind of computer with some functional equivalence to a universal Turing machine. It was Newell, along with his longtime collaborator, Herbert Simon, who asserted that physical symbol systems have the necessary and sufficient means for general intelligence (Newell & Simon, 1976). Pylyshyn (1999) suggests that it is the task of cognitive science to discover the details of the mechanisms that support mental computation; i.e., to determine what kind of computer the human mind is. Thus, it is the task of cognitive science to discover the cognitive architecture of the mind.

A cognitive (or, as it is sometimes referred, functional) architecture is a bridging of the biological and cognitive levels of description, whereby the computational primitives of human cognition are defined. Cognitive architectures are analogous to

computer architectures in that both describe the physical constraints (e.g., memory capacity) on the algorithms that run in the architecture. Thus, the functional details of the cognitive architecture are constrained by the underlying physical system implementing the architecture. According to Pylyshyn (1998), facts about the knowledge level depend only on the functional details of architecture, and not on the details of how the architecture is implemented. This paper devotes attention to a particular cognitive architecture: ACT-R.

### ACT-R

ACT (adaptive control of thought) theory was originally proposed by John Anderson in 1976. ACT was essentially a marriage of Anderson and Bower's existing model of declarative memory, HAM (1973), and a production system based on Newell's proposal (1973b). ACT evolved into ACT\* (1983), then into ACT-R in 1993. ACT-R has undergone several significant revisions. In 1998, the release of ACT-R 4.0 coincided with the publication of the book "The Atomic Components of Thought" (Anderson & Lebiere, 1998). In the opinion of Anderson and Lebiere (1998), ACT-R 4.0 was the first version of ACT-R to legitimately realize "Newell's dream" of a unified theory of cognition.

ACT-R is a cognitive architecture implemented in LISP. ACT-R is also a theory of cognition. According to the designers of ACT-R, the atomic components of thought are chunks, which exist in declarative memory and production rules, which exist in procedural memory. A chunk is an independent pattern of information corresponding to a thing that one can be aware that one knows. Each chunk consists entirely of several slots

(variables) with associated values. The slot values are either chunks themselves, or atomic features. The designers of ACT-R assert that chunks should have, on average, three or four slots, one of which must be an ISA slot. The ISA slot determines to what ontological type a particular chunk token corresponds (e.g., john841 *is a* person chunk). Respecting George Miller's "magic number", chunks should have no more than seven (plus or minus two) slots (Miller, 1956).

Chunks originate primarily from two sources: perceived objects in the environment and records of solutions to past problems. Production rules specify how to retrieve and use chunks to solve problems. Each production rule consists of an if-then condition-action pair. The ISA value of chunks plays an integral role in determining whether a chunk matches conditions in the production. New production rules can be generated from chunks in memory via a process called production compilation.

In this paper, chunks will be written in the format [Chunk-Name: ISA Chunk-Type; slot-1-label slot-1-value; slot-2-label slot-2-value; etc.]. For example, an ACT-R chunk representing the knowledge that  $3 + 4 = 7$  could be encoded [Fact3+4: ISA addition-fact; addend1 Three; addend2 Four; sum Seven].

### *Newell's Criteria Revisited*

I suggest that whether ACT-R realizes "Newell's dream", or not, is unclear. As of version 4.0, ACT-R has been used to produce successful models of the cognitive phenomena in the first two groups of areas identified by Newell (i.e., problem solving, decision making, routine action, memory, learning, and skill). And, with the development of ACT-R 5.0 and a perceptual-motor system, ACT-R/PM (Byrne & Anderson, 2001), the

areas in the third group (perception, and motor behaviour) have now been added to the list of ACT-R's successes. I take the adequacy of ACT-R models of the remaining areas identified by Newell to be, at a minimum, somewhat uncertain.

The question remains: in what way is it unclear whether ACT-R realizes Newell's dream. The problem is with respect to what counts as meeting Newell's criteria. Newell asserts that a unified theory of cognition should have a single story to tell about the various areas he identifies (Newell, 1990, p. 15), which is consistent with all previously existing psychological knowledge of each area of study (Newell, 1990, p. 16). Interestingly, from a modeling perspective, unifying theories can take two distinct forms, with two different criteria. I will define here my weak and strong criteria for a universal theory of cognition qua modeling toolkit. The weak criterion is that a unified theory of cognition should provide a common set of tools and principles such that for any psychological phenomenon a model that accounts for the phenomenon can be created. This criterion does not imply that any pair of cognitive phenomena can be reconciled within a single model existing within such a unified theory of cognition. Hence, the strong criterion is that a unified theory of cognition should provide a common set of tools and principles such that a single model that accounts for every psychological phenomenon can be created. Simply, the strong criterion is met if and only if (in principle) a single model of the whole mind can be created. Additionally, it should be noted that I use the term "model" in the sense of a tokened instance of a type defined within the parameter space of a particular cognitive architecture. ACT-R has enjoyed a

great deal of success in meeting the weak criterion<sup>1</sup>. However, I do not know of any attempts to satisfy the strong criterion, nor, have I come across any explicit reference to a distinction between the weak and strong criteria, in the ACT-R literature.

I suggest that in order to realize the spirit of Newell's dream, the strong criterion must be met. Cognitive science is about understanding the mind as a unified single entity, not as a finite collection of phenomena, or abilities. Hence, to be able to build a model of an entire mind should be our goal.

*Computational Primitives: The Atomic Components of Thought*

A computational primitive is a non-decomposable atom of thought. Although, there is in principle a distinction between an atom of thought and an atom of knowledge, there is no such distinction in ACT-R. Chunks are the atoms of declarative knowledge, and productions are the atoms of procedural knowledge. In ACT-R, declarative memory is an associative network of persistent chunks, each of which has a particular activation level. When the cognitive system engages in a cognitive process, the very same chunks that exist in memory are the ones recruited to take part in the process. Thus, according to ACT-R, the grain-size of one's basic level of knowledge of the world is identical to the grain-size of knowledge as it is used in active thought.

ACT-R is designed to model individual experimental tasks. Anderson and Lebiere write: "every ACT-R model corresponds to a subject performing in some

---

<sup>1</sup> Visit <http://act-r.psy.cmu.edu/publications/> for list of publications on ACT-R, a majority of which present ACT-R models of cognitive phenomena.

particular experiment” (1998, p. 15); and, “most ACT-R models assume a system that starts out with substantial *relevant* knowledge, as is the case for the typical undergraduate subject” (1998, p. 16, my emphasis). If what is meant by relevant, is “task specific”, then there may be issues with respect to the nature of persistent knowledge in declarative memory? A principle that contributed to the development of knowledge representation in ACT-R is that for every known entity in the world, there should be only one chunk in memory. In reference to an early version of ACT where nodes can be interpreted as chunks, Anderson wrote: “these memory systems assume only one node per individual, reflecting the spatio-temporal continuity of that individual” (1977, p. 430). Additionally, “there are no special ‘subtoken’ nodes connected to the individual to represent different aspects of the individual” (Anderson, 1977, p. 430). These constraints are presumably motivated by both theoretical and practical considerations. From a theoretical perspective, as suggested, Anderson seems committed to a sort of isomorphism between ontologically distinct entities in the world and representations of them in the mind. From a practical perspective, it is much easier to retain consistency in a knowledge base when there is no redundancy in representation.

The one-entity-one-chunk constraint seems to eliminate the possibility of distinct task specific chunks. Therefore, if ACT-R models are to scale up to modeling the entire mind, chunks will have to encode information in a neutral manner that can be employed during a variety of tasks. Thus, chunks should not admit of context specific features, nor, should they admit of vague values. The reason for this latter condition is that it can be argued that vague predicates have no meaning outside of particular contexts. In addition, even within a given context a vague predicate can fail to have determinate truth-values.

Although the relevance of a predicate's truth conditions to its meaning is a matter of some debate, I will assume that the two notions are, at a minimum, related.

This raises the question, why is it the case that vague predicates have been employed in atomic representations within ACT-R without difficulty, so far. Although, vagueness is different from context dependence (e.g., "Tim is above average in height" is context dependent, but not vague), what a vague predicate asserts of an entity depends on context (e.g., "Tim is tall" makes a different claim about Tim's height in the context of a set of basketball players as compared to a set of racehorse jockeys). Therefore, the truth-values of vague predications depend on more than just the entity of which the predicate is applied. It depends on, in the very least, an implied comparison set. Such an implied comparison set is furnished by the particular context in which the given ACT-R model exists. However, chunks that contribute to a general knowledge base ought not to be context dependent, if there is to be no redundancy in representation of knowledge (i.e., the casting of some fact about the world as multiple chunks, one for every possibly relevant context would entail some redundancy).

#### *The Chunk Capacity Problem*

There is an issue related to which slot values ought to be associated with a particular chunk. It is obvious that my most basic knowledge of a whole entity includes more than a few features. Even of some stranger I observe walking down the street, I can perceive: Their apparent gender, ethnicity, mood, their basic physical features, what they are wearing, of whom their appearance reminds me, and so on. There are clearly more

than seven plus or minus two features I may need to encode in my person chunk type<sup>2</sup>. The problem is greater for people of whom I actually have a plethora of knowledge.

ACT-R provides two mechanisms for potentially avoiding this problem. The first is inheritance. When defining a chunk type, a superordinate chunk from which the new chunk inherits slots, may be specified. For example, my chunk token representing my cat Cougar could be of the species chunk type domestic cat, which inherits from the family type Felidae, which inherits from the order type Carnivora, which inherits from the class type Mammalia, and so on. By this system, each chunk type in the hierarchy would specify up to seven slot values representing things people know about examples of those types. Thus, every chunk can have up to  $7n$  slots, where  $n$  is the chunk's rank in the hierarchy. This does not necessarily violate Miller's magic number constraint. For example, productions could be limited to specifying a maximum of seven slot values per chunk for the purposes of goal matching and memory retrieval etcetera. Thus, only the seven most relevant aspects of a chunk would be featured in a particular cognitive process. This appears to be a clever way to accommodate context in ACT-R models.

The problem with this solution is the nature of inheritance in ACT-R. Each chunk may inherit from only one superordinate chunk. Thus, every chunk type can participate in only one conceptual hierarchy. The effect of this is that the definition of each chunk type is limited to only one epistemic stance. For example, if my "Cougar" chunk inherits from my domestic cat chunk type, which would place it in a biological hierarchy, it

---

<sup>2</sup> I will rhetorically refer to myself as an ACT-R model of a human throughout this paper.

cannot also inherit from my pet chunk type, which would place it in a different (perhaps functional) hierarchy. Pace the standard ACT-R doctrine, it seems reasonable that an atomic component of knowledge ought to be able to participate in a variety of conceptual hierarchies, one for each unique epistemic perspective that can be taken of the thing represented by that atomic component of knowledge. The utility of this idea evidenced by the variety of computer programming languages that permit multiple-inheritance in object type definition such as C++ and (the increasingly popular) python programming language<sup>3</sup>.

The second mechanism provided by ACT-R for avoiding the problem of trying to accommodate all knowledge of a thing into a single chunk, is to create chunks that encode relational knowledge. Any logical expression can be encoded in a chunk. For example, “John loves Mary”, might be encoded by the chunk [John-loves-Mary: ISA relation; relation love; agent John; theme marry]

The principle problem with this solution is the proliferation of chunks in declarative memory. That is, there would be thousands, if not millions of relational chunks in memory, one for every relation that one had ever entertained. This is a problem, because not all of these relations are independent of one another. Specifically, changes to some relations should affect others. For example, if I learn that that John has died, every (or at least most) chunk(s) relevant to him must be updated. Unfortunately, ACT-R does not provide an easy way to modify all chunks relevant to a particular other chunk; in ACT-R, qualitative knowledge is only modified by production firings. To

---

<sup>3</sup>See <http://www.python.org/>

update every chunk of which John is a slot value, a separate retrieval attempt for every combination of chunk type and slot in which John could potentially occur must be made. Additionally, there would probably have to be a unique production for updating each kind of chunk. For example, chunks [John-wrote-the-book-10-ways-to-catch-a-sparrow: ISA authorship-relation; agent John; theme '10 way to catch a sparrow'], [John-is-married-to-Hilary: ISA marriage-relation; person1 John; person2 Hilary], and, [John-is-smelly: ISA an-odour-fact; individual John; scent-type smelly] should be updated differently, if at all. To make matters worse, this speculation about the means of updating chunks is not even possible! In the case of real minds, new knowledge of this sort is, more-or-less, seamlessly integrated into the knowledge base. In contrast, ACT-R productions fire serially, and require a minimum of 50 ms each to fire. Thus, it would take an ACT-R model of the mind hours to complete firing the potentially thousands of productions required to update its knowledge base. ACT-R is not designed to operate over a large knowledge base, in real time.

#### *Dreyfus, GOFAI, and ACT-R*

In general, ACT-R falls victim to many of the criticisms made against traditional (good old fashioned) A.I. systems. Limiting current ACT-R models to individual experimental scenarios is akin to limiting Terry Winograd's SHRDLU (Winograd, 1972) to interacting with a drastically circumscribed blocks micro-world. Hubert Dreyfus argued that SHRDLU failed as a model of understanding because understanding is a concept whose meaning depends on a vast network of other human concepts, which are absent from the block micro-world (Dreyfus, 1979). Dreyfus writes:

In our everyday life we are, indeed, involved in such various 'sub-worlds' as the world of the theatre, of business, or of mathematics, but each of these is a 'mode' of our shared everyday world. That is, sub-worlds are not related like isolable physical systems to larger systems they compose; rather they are local elaborations of a whole which they presuppose... Since,..., micro-worlds are not worlds, there is no way they can be combined and extended to the world of everyday (1979, p. 151).

This criticism is consistent with the theme of my weak and strong criterion distinction. No collection of ACT-R models, each performing in its own micro-world, will collectively inform us about how a single human mind accomplishes each of these tasks in our 'everyday' world.

Dreyfus was an early advocate of the notion of situated cognition, which continues to be researched by the likes of Andy Clark (1997), among others. The similarity in structure of ACT-R chunks and Marvin Minsky's frames makes Dreyfus' attack on Minsky's Frame theory (1974) relevant to the current topic:

No piece of equipment makes sense by itself... What makes an object a chair is its function, and what makes possible its role as equipment for sitting is its place in a total context... There is no argument why we should expect to find elementary context-free features characterizing a chair type, nor any suggestion as to what these features might be. They certainly cannot be legs, back, seat, and so on, since these are not context-free characteristics defined apart from chairs which then 'cluster' in a chair representation (Dreyfus, 1979, p. 163-4).

Dreyfus is questioning if it is even in principle possible provide adequate formal criteria for our everyday human concepts. This scepticism strikes directly at the basis of ACT-R's knowledge representation scheme.

### A Different Kind of Cognitive Architecture

I have argued that a network of ACT-R chunks cannot form the basis for humans' basic level of knowledge of the world. However, it seems entirely plausible that when we actually engage in consciously accessible cognitive processes, ACT-R has the grain-size right, vagueness and all. What I suggest is that ACT-R is a good candidate for a model of high-level cognitive processing which sits on top of a low-level dynamical system responsible for encoding knowledge in a holistic manner.

A researcher who has taken a great interest in the prospect of modal knowledge representation is Lawrence Barsalou also known for his work on ad-hoc category representations (Barsalou, 1983). Barsalou suggests that perceptual symbol systems via his situated simulation theory underlie much of our conceptual knowledge (Barsalou, 1999; Barsalou, 2003; Barsalou, Simmons, Barbey, Wilson, 2003). Barsalou writes, "A concept is not a single abstracted representation for a category, but is instead a skill for constructing idiosyncratic representations tailored to the current needs of situated action" (2003, p.521). An example to which Barsalou theory speaks is the use of visual imagery.

In 1978, Steven Kosslyn published research on visual imagery (Kosslyn, et al., 1978). Briefly, participants studied a map with various landmarks; with their eyes closed, they were asked to imagine the map and focus on a particular landmark; a different landmark was named, and the participant was instructed to hit a button once they could 'see' the second landmark. It was found that the time taken to 'see' the second landmark was proportional to the distance between the landmarks on the map. Thus, Kosslyn concluded that people were using mental images, via the 'mind's eye' to perform the task. These findings are consistent with the idea that knowledge about visual information is

stored experientially in the visual cortex. According to Barsalou's theory, when the subject is asked a question about the map, they simulate seeing the map and respond in a manner similar to how they would if they were viewing the original map. They do not rely on explicit symbolic long-term memory.

Pylyshyn responded to the results of Kosslyn's classic 'mind's eye' experiment, by suggesting that spatial information can be encoded symbolically (Pylyshyn, 1999). Pylyshyn argued that the correlation between time and distance in the results could merely be an artifact of the 'imagining' process and not due to operations on a spatial mental representation of the map in the participants' minds. This argument is supported by the fact that in a response study, Pylyshyn was able to show that people could answer questions about aspects of a map instantaneously, if the subjects were not asked to imagine traversing the map (Pylyshyn, 1999). This suggests that it is at least possible that a declarative symbolic representational system underlies one's ability to engage in mental imagery.

Pylyshyn's counter-example suggests only that people have the capacity to encode visual information in a declarative manner, and not that people systematically do so. The use of examples of visual imagery may be misleading. People are largely very good at interpreting visual information declaratively. However, a much smaller proportion of the general population can do so with auditory information. For example, consider the case of pitch comparison. It is not obvious that people who do not have any musical training have very detailed declarative knowledge of the music to songs they know. When asked to determine whether the note associated with "dash" in the song jingle bells is higher or lower than the note associated with "sleigh", most people have no alternative but to

“simulate” singing the song to themselves and make an online comparison between the two notes via auditory imagery (Zatorre, 2004). Barsalou would argue that, fundamentally, our knowledge of the song jingle bells resides in our auditory sensory system, and not in an amodal memory system such as ACT-R’s declarative memory system.

Barsalou’s theory that all conceptual knowledge is modal may be too extreme. However, is it possible that a symbolic representation underlies my ability to sing a song to myself, but is not flexible enough for me to directly access information about the notes associated with each word in the song? What seems more likely is that people can know jingle-bells in at least two different ways. The non-musically trained know it only via the experience of hearing and singing it. The musically trained can have explicit knowledge of the musical notation associated with the song. Interestingly, it is well known that Beethoven was deaf during the time period that he composed some of his best known works. Clearly, Beethoven had never heard his later compositions. However, he may very well have been able to simulate the experience of hearing his music by directly translating his explicit knowledge of the sheet music associated with a given work to a phenomenal (internal) experience of music, via the mind’s ear.

I suggest that Barsalou’s situated simulation theory can be reconciled with ACT-R. Chunks would be abandoned as a form of atomic knowledge representation (as opposed to atomic thought representation as discussed above). Modular perceptual symbol systems would take on the burden of encoding knowledge at the lowest level relevant to cognitive psychological research. The perceptual symbol systems would be augmented with a sort of amodal knowledge system which underlies people’s declarative

knowledge base. ACT-R would sit on top of these perceptual symbol systems. In a given context the perceptual symbol systems and the amodal system would generate, on the fly, a set of temporary task specific chunks and productions which would exist only while the system is attending to a particular cognitive task. The temporary chunks generated would only serve as atomic components of (current) thought, and no longer as atomic components of (persistent) knowledge.

This proposed unification of Barsalou's situated simulation theory with ACT-R is an example a hybrid cognitive architecture which I take to be necessary for building models of the whole mind. There is a sort of autonomy of high-level cognitive processes from the underlying physics. However, the autonomy of a transient ACT-R model of a task is due to the fact that it exists only as an abstraction from the underlying knowledge base of the system as a whole.

### ACT-R 6

ACT-R is an evolving architecture. It has undergone significant revisions and changes over the years, and soon, the next version, ACT-R 6 will be released. ACT-R 6 will incorporate several changes to ACT-R 5. One change that is relevant to my concern with the current version of ACT-R is increased modularity (Bothell, Byrne, Lebiere, Taatgen, 2004). In previous versions of ACT-R, all cognitive processes were implemented as productions operating on the contents of the goal buffer, which is a place holder for a single active declarative chunk. This necessarily serialized all high level cognition. ACT-R 6 will allow for multiple modules each of which will have its own declarative chunk buffer. This change should increase the amount of parallelism in ACT-

R somewhat. However, more significantly, this change allows for a degree of autonomy of some cognitive processes (e.g., visual perception) from operations on the goal buffer. This decentralisation is a necessary step in adjusting the ACT-R paradigm towards the hybrid architecture that I have suggested.

This current trend in the evolution of ACT-R may naturally lead it towards solutions to the problems I have identified with the architecture, which may be similar to those that I have suggested. Whether future changes, less radical than what I have proposed will suffice, is yet to be seen.

#### References

- Anderson, J. R. (1977). Memory for information about individuals. *Memory and Cognition*, 5, 430-442.
- Anderson, J. R., & Lebiere, C. (Eds.) (1998). *The atomic components of thought*. Mahwah, NJ: Lawrence Erlbaum.
- Barsalou, L. W. (1983). Ad hoc categories. *Memory & Perception*, 11, 211-227.
- Barsalou, L. W. (1999). Perceptual symbol systems. *Behavioral and Brain Sciences*, 22, 577-609.
- Barsalou, L. W. (2003). Situated simulation in the human conceptual system. *Language and Cognitive Processes*, 18, 513–562.
- Barsalou, L. W., Simmons, W. K., Barbey, A. K., and Wilson, C. D. (2003). Grounding conceptual knowledge in modality-specific systems. *TRENDS in Cognitive Science*, 7, 84-91.

- Bothell, D., Byrne, M. D., Lebiere, C., & Taatgen, N. A. (2004). *ACT-R 6 proposals*. Retrieved from <http://act-r.psy.cmu.edu/act-r6/ACT-R6proposal.pdf>
- Clark, A. (1997). *Being there: Putting brain, body, and world together again*. Cambridge, MA: The MIT Press.
- Dawson, M. R. W. (1998). *Understanding cognitive science*. Malden, MA: Blackwell.
- Dreyfus, H. L. (1979). From micro-worlds to knowledge representation: AI at an impasse. In J. Haugeland (Ed.), *Mind design II* (pp. 143-182). Cambridge, MA: The MIT Press.
- Kosslyn, S. M., Ball, T. M., & Reiser, B. J. (1978) Visual images preserve metric spatial information: Evidence from studies of image scanning. *Journal of Experimental Psychology: Human Perception and Performance*, 4, 46-60.
- Laird, J. E., & Rosenbloom, P. S. (1996). The evolution of the soar cognitive architecture. In Steier, D. M., & Mitchell, T. M. (Eds.), *Mind Matters: A tribute to Allen Newell* (pp. 1-50). Mahwah, NJ: Erlbaum.
- Miller, G. A. (1956). The magical number seven, plus or minus two: Some limits on our capacity for processing information. *Psychological Review*, 63, 81-97.
- Minsky, M. (1974). A framework for representing knowledge. In J. Haugeland (Ed.), *Mind design II* (pp. 111-142). Cambridge, MA: The MIT Press.
- Newell, A. (1973a). You can't play 20 questions with nature and win: Projective comments on the papers of this symposium. In W. G. Chase (Ed.), *Visual Information Processing* (pp. 283-310). New York: Academic Press.
- Newell, A. (1973b). Production systems: Models of control structures. In W. G. Chase (Ed.), *Visual Information Processing* (pp. 463-526). New York: Academic Press.

- Newell, A. (1990). *Unified theories of cognition*. Cambridge, MA: Harvard University Press.
- Newell, A. & Simon, H. (1976). Computer science as empirical enquiry: Symbols and search. In J. Haugeland (Ed.), *Mind Design II* (pp. 81-110). Cambridge, MA: The MIT Press.
- Pylyshyn, Z. W. (1998). Introduction: Cognitive architecture and the hope for a science of cognition. In Z. W. Pylyshyn (Ed.), *Constraining cognitive theories* (pp. 1-7). Stamford, CT: Ablex Publishing Corporation.
- Pylyshyn, Z. W. (1999). What's in your mind. In E. Lepore & Z. Pylyshyn (Eds.), *What is cognitive science?* (pp. 1-25). Malden, MA: Blackwell
- Winograd, T. (1972). Understanding natural language. *Cognitive Psychology*, 1, 1-191.
- Zatorre, R. (November, 2004). Processing of music and speech by the human auditory cortex: Neuroimaging evidence. *Carleton Cognitive Science Research Seminar*. Ottawa, ON.

## MALTA: ENHANCING ACT-R WITH A HOLOGRAPHIC PERSISTENT KNOWLEDGE STORE

Previously published as Rutledge-Taylor, M. F. & West, R. L. (2007). MALTA: Enhancing ACT-R with a holographic persistent knowledge store. In D. S. McNamara & J. G. Trafton (Eds.), *Proceedings of the 29th Annual Conference of the Cognitive Science Society* (pp. 1433-1438). Nashville, TN: Cognitive Science Society.

### Contribution

This paper outlines a framework, MALTA, for augmenting ACT-R DM with a supplemental long-term knowledge store based on DSHM. This paper supports Rutledge-Taylor (2005), by making specific recommendations for how ACT-R DM could be enhanced in such a way that it could avoid the problems described in Rutledge-Taylor (2005).

### Abstract

There is a concern that ACT-R (Anderson & Lebiere, 1998), and other cognitive architectures do not adequately account for long-term memory (Schultheis, Barkowsky, & Bertel, 2006). These architectures, while ideal for modeling single tasks, seem to be incapable of preserving a task neutral knowledge store, which persists between tasks (Rutledge-Taylor, 2005). A dynamical holographic associative memory system, DSHM, based on Jones and Mewhort's model of the lexicon, BEAGLE (2007), is presented. A suggestion for a new cognitive architecture, MALTA, which combines elements of ACT-

R and DSHM is described. Arguments for why this new architecture addresses some concerns with ACT-R's declarative memory system are offered.

### Introduction

Modeling memory is not new; detailed descriptive models of memory have existed since the 1960s. Atkinson and Shiffrin (1968), Craik and Lockhart, (1972), Tulving (1972), and, Baddeley and Hitch (1974), among others, are credited with providing the first early influential models of memory. A more recent phenomenon is the formalization of memory models using computers. Since memory does not exist in isolation from other components of the human mind, it is natural to embed memory within a larger scheme for modeling all of cognition. The concept of the cognitive architecture arose from this rationale.

A cognitive architecture is a means for formalizing the principles that underlie one's theory about how the mind works (Newell, 1990). It specifies the structure of the mind, including how information is gathered, processed, and used to perform tasks. There are several advantages to implementing a cognitive model. Using the architecture, models can be built which can then be used to simulate of human performing tasks. The resulting simulation data can be compared to human experimental data, which allows for both the evaluation of the particular model, and the formalizations specified by the architecture itself.

Three of the most popular cognitive architectures are ACT-R (Anderson & Lebiere, 1998), SOAR (Laird, Newell & Rosenbloom, 1987; Newell, 1990), and EPIC (Kieras & Meyer, 1997). They are all software implemented formal theories of how the

mind processes information; however, they each make different commitments to how this occurs in detail. Of these, ACT-R has the best developed and well tested model of the human declarative memory system and is also highly consistent with mainstream theories about declarative memory from cognitive psychology.

The declarative memory system of ACT-R (DM) arose out of a need to account for the results of experiments on human memory generated by experimental psychologists. DM is also a necessary component of accounts of most experiments on cognition, due to the need to account for participants' memory of experimental instructions, and relevant general background knowledge. In developing an ACT-R model, a knowledge base relevant to the experimental task, is usually provided to the model in advance of a simulation to represent knowledge held by the subject before beginning the experiment. During the experiment the model typically learns by creating new chunks and adjusting the relative activation of existing chunks (Anderson & Lebiere, 1998). This general scheme, and variations on it, has been very successful for modeling the sorts of tasks given to participants in psychological experiments related to memory and learning (Anderson & Lebiere, 1998).

However, a unified theory of cognition ultimately needs to go beyond modeling individual tasks done in isolation (Newell, 1990). It must also account for how it is that we, as intelligent agents, accumulate a vast amount of world knowledge that we apply in a context specific manner to the various tasks that we perform throughout our lives. In order to make this distinction clear we will refer to the knowledge that we carry around outside of a particular context as *persistent knowledge*, and the specific knowledge needed for a particular task as *task knowledge*. Task knowledge is a local manifestation

of persistent knowledge that has been retrieved or highlighted to make it available for the task. ACT-R is a very good system for modeling how people use and manipulate task knowledge, but there is very little work focusing on the management of persistent knowledge. One approach that has been used for modeling how people process the enormous database that makes up persistent knowledge is the use of holographic memory models.

The principle purpose of this paper is to present our Dynamically Structured Holographic Memory System (DSHM), a means for modeling persistent knowledge and to suggest how it could be used to augment ACT-R so as to produce a larger scale architecture suitable for modeling human cognition as we have characterized it. The format for the remainder of this paper is as follows: we will describe the holographic memory system developed, DSHM, based on the lexical representation system BEAGLE (Jones & Mewhort, 2007); and, describe how a hybrid system, MALTA, combining DSHM and ACT-R, can in principle address the problems with ACT-R we have described.

### Holographic Associative Memory

#### *BEAGLE*

DSHM is based on Jones and Mewhort's BEAGLE (Bound Encoding of the Aggregate Language Environment) model (Jones & Mewhort, 2007). The BEAGLE model learns a lexicon by creating holographic representations of words. The resulting memory system is capable of accounting for a variety of psychological data pertaining to the lexicon.

The BEAGLE system takes as input, sentences, one at a time. Every word is represented internally, by the system as an item composed of two large vectors of numbers. One, called the environmental vector, uniquely codes the word and never changes. The other, called the memory vector, encodes information about which other words co-occur with the given word in the sentences imported into the system. This memory vector is modified in two ways. For every word in a sentence imported into the system, the sum of the environmental vectors of every other word is added to the given word's memory vector. This imbues the given word with traces of every word it has co-occurred with, but in a manner which is insensitive to the relative order of the words in the given word's environment. In order to preserve this order information a subsequent procedure is performed. The details of this rather complicated algorithm will not be described in full here. However, a key feature is that Jones and Mewhort make use of Tony Plate's holographic technique for binding vector representations of words together via circular convolution (Plate, 1995). Binding two vectors in this manner results in a vector with the same length as each of the original vectors. This allows for the recursive binding of lists of words.

Based on these two methods of encoding information about a word's neighbourhood, BEAGLE is able to develop a rich lexicon if provided a large enough corpus. After training, the memory vectors of words in the system can be analysed. The Euclidean distance between any two words is correlated with the difference in meaning of the two words. With sufficient training synonymous words will eventually converge to having identical memory vectors; the words appear in all of the same language contexts. Any difference in the memory vectors of near synonyms is indicative of idiosyncrasies in

which contexts each word does and does not appear. For example, ‘big’ and ‘large’ appear in many of the same contexts and are often interchangeable. However, ‘large’ cannot be substituted for ‘big’ in “big sister”. Thus, the two words are only near synonyms, and this would be reflected in BEAGLE by slight differences in their memory vectors.

In their published results, Jones and Mewhort (2007) used pairs of 2048 element vectors to represent words, and provided BEAGLE with a corpus of text approximately equal to what an average undergraduate student will have read in his or her lifetime. Words learned by the system distributed themselves over the state-space defined by their memory vectors in such a way that categorically similar words grouped together in a hierarchical manner: within the lexicon, nouns and verbs occupied distinct areas; within nouns, animal words will cluster separately from vehicle words; and, within animal words, fish and birds cluster distinctly, and so on. Thus, BEAGLE effectively solves a constraint satisfaction problem, where all of the words in a lexicon must be distributed in a finite multi-dimensional state space where the semantic relationships between the words are preserved in the distances between words in the space. See Jones and Mewhort (2007) for a detailed account of how this structure within BEAGLE can be used to account for the results of various experiments on the lexicon, e.g., typicality judgments (Rosch, 1975).

BEAGLE can be understood as a pattern recognition system. The grammatical features, and more generally, the lexical features of English that it learns are discovered only by exploiting statistical regularities in the text imported into the system. As such, if presented with a sentence (or, any other string of words) with missing words, BEAGLE

can be used to complete it. For example, if “he \_\_\_\_ to the highway” is presented, where ‘\_\_\_\_’ indicates a missing word, the system will nominate words like ‘went’ and ‘drove’ as those most suitable for filling in the blank. However, if “he \_\_\_\_ to the audience” is presented, ‘spoke’ would be the top candidate generated by BEAGLE. A key aspect of this ability of BEAGLE is that it is very sensitive to context. The two sentences above differ only by a single word. However, it is the presence of ‘highway’ in the first and ‘audience’ in the second that allows BEAGLE to hone in on a unique, small set of substitution candidates in each example.

The lexicon that BEAGLE develops is referred to as holographic due to the fact that it has some high-level features of holograms. Holograms compress a three-dimensional scene onto a two-dimensional film; and, every part of the holographic film stores information about the entire scene. Similarly, BEAGLE takes some subset of the natural language English, which has no defined dimensionality and is perhaps best understood as infinitely dimensioned, and represents it in 2048 dimensions, the length of the vectors used to represent words. Additionally, every word in the BEAGLE lexicon stores information about the entire language. Specifically, the lexical information stored in the memory vector of each word is precisely the sum of all the contexts, properly encoded, in which it has appeared. Interestingly, the memory vectors of ubiquitous grammatical words such as the article ‘the’ store very little information due to the fact that the diversity of lexical contexts in which they appear blur the contributions of each individual context. Words that appear in very specific contexts, such as ‘Pharaoh’ tend to be very informative about the contexts in which they appear. This is intuitively correct. The reader may recognize that the word ‘the’ does not bring to mind very many specific

related concepts, whereas 'Pharaoh' immediately brings to mind ideas such as the great pyramids, and ancient Egypt.

It should be noted that the neuroscientist, Karl Pribram was probably the first researcher of human cognition to recognize that the brain has some of these features in common with holograms (Pribram, 1971; 1991).

*DSHM (Dynamically Structured Holographic Memory System)*

Jones and Mewhort's BEAGLE is a very important first step in understanding human memory and human cognition in general. BEAGLE is a powerful model of the lexicon. It demonstrates that knowledge of the referents of words is not necessary for many features of the lexicon to develop during the learning of a language. These features are extracted from only the statistical features of the (written) language.

Our claim is that the mechanisms employed by BEAGLE can be generalized such that they apply to information processing generally, and not to the processing of language in particular. BEAGLE takes as input sentences, which can be interpreted as ordered sets of words. DSHM operates on objects, which like the items in BEAGLE consist of an environmental vector and a memory vector. However, objects also consist of a set of components, which are either an ordered set of objects, an unordered set of objects, or the empty set (in the case of the atomic objects of the system). An ordered relationship between objects  $x$  and  $y$  can be variously interpreted as:  $x$  is to the left of  $y$ ;  $x$  is superordinate to  $y$ ; or,  $y$  is a token of the type  $x$ , etc. Thus, objects have recursive compositional structure over other objects. Complex objects can be derived from their

parts, but exist as separate objects in the system to reduce the computational load that would be incurred in repeatedly generating them from the atomic objects, when needed.

ACT-R chunks are hierarchically structured. Each ACT-R chunk type consists of one or more slots, which act like variable names. Each instance of a chunk consists in the pairing of each slot with a value, which can be either an atomic object of the system or another chunk. An ACT-R chunk can thus be interpreted as an unordered set of ordered pairs of objects, where each pair consists of a slot and a value. The set of pairs is unordered, because the order in which the slots of an ACT-R chunk are defined is not relevant to the representational content of the chunk. The pairs themselves are ordered, because for each slot/value pair, which object is the slot and which is the value is relevant. It should be noted that in DSHM there is no privileged distinction between a slot and a value as kinds. That which can appear as a slot can also appear as a value, and vice-versa—just as BEAGLE is able to discover the syntactic categories of English, DSHM should similarly discover any regular type/token distinctions etc. in the data to which it is exposed.

DSHM is very flexible with respect to how to encode information. For example, mathematical knowledge in DSHM can be encoded similarly to the classic ACT-R example: [isa:addition-fact addend1:three addend2:four sum:seven]. Here, a colon indicates that the items to the left and right bear an ordered relationship to one another, and a space indicates that the items to the left and right bear an unordered relationship to one another. By default, spaces delimit ordered sets. However, DSHM affords the possibility of defining this chunk differently: [isa:addition-fact [3 4]:7]. Rather than using slots and values for all of the content of the object, only the type of chunk is

defined in this way. The remainder, [3 4]:7, can be interpreted as, “the addends 3 and 4, bear a relation, sum, to the number 7”. Internally, the object [isa:addition-fact [3 4]:7] is composed as follows: the objects ‘isa’ and ‘addition-fact’ are bound together to form the complex ordered object [isa:addition-fact]; the objects ‘3’ and ‘4’ are bound together to form the complex unordered object [3 4]; the objects ‘7’ and [3 4] are bound together to form [3 4]:7; and finally, the objects [isa:addition-fact] and [3 4]:7 are bound together. A feature of this representation is that, because [3 4] is unordered, it is identical to [4 3], and so, the system automatically knows that ‘4+3=7’, after learning ‘3+4=7’. Other ways of representing this mathematical knowledge can be devised. As with ACT-R, comparisons to human data on math competence (e.g., from Mauro, LeFevre & Morris, 2003), would help determine what representational scheme best models human knowledge of mathematical facts.

#### *Retrieval From Memory*

The retrieval of information from DSHM can be accomplished in two ways. First, the environmental vector of an object can be presented to the DSHM, and the objects with memory vectors that resonate with (are similar to) that vector will be retrieved. This is effectively asking the system for objects that have co-occurred with the given object. Second, an incomplete complex object can be presented to the DSHM. The system uses a rather complex algorithm to decode which objects best fill the missing parts of the object. For example, if presented with [isa:addition-fact [4 ?]:7], where ‘?’ indicates a missing object, the system will first nominate completions of [4 ?]. A list of pairs consisting of ‘4’ and one other object will be generated. These pairs will be based,

roughly, on asking ‘4’ who occurs most with it. Second, this list of pairs is presented to ‘7’, and ‘7’ is asked which of those object pairs often precedes it. This will drastically, reduce the number of objects remaining in the list by discarding pairs like [4 o’clock], and [connect 4] etc. Next, the candidates for the completion of [[4 ?]:7] are presented to the complex object [isa:addition-fact], where [[4 3]:7] will resonate best.

This object structure provides DSHM with flexibility not available to BEAGLE. DSHM is able to operate on sentences, just as BEAGLE does. To model sentence processing using DSHM, each sentence would be represented as an object with an ordered set of parts consisting of atomic objects representing each word. However, it is not necessary that objects representing words be atomic objects. Each word could have as parts an ordered list of atomic (objects representing) phonemes. The ability to create objects with hierarchical structure allows the study of top-down versus bottom-up processing. For example, DSHM can be used to investigate phoneme restoration effects by manipulating whether words with deleted phonemes are presented alone or in the context of a sentence (Samuel, 1981). For example, one measure could be whether the phoneme missing in the word “b\_g”, where ‘\_’ indicates where a phoneme has been deleted, will be restored differently in the contexts, “b\_g” alone, “the boy gave her a b\_g for her groceries”, and “the boy gave her a b\_g for her gecko”. If trained on a corpus where the words “big”, “bag”, and “bug” occur, with “big” appearing more frequently than the other two words, DSHM ought to predict that the “big” would be the top candidate in the first case. However, in the cases where the incomplete word occurs in the context of a sentence, the system subsequently evaluates how well the candidate words “big”, “bag”, and “bug”, match the context of the rest of the sentence. If “bag” is

much more likely to occur in the context of “groceries” than is “big”, then “bag” could get a sufficient boost to move it ahead of “big” in the subsequent re-ranking of the candidate words, in the second case listed above. Whether this would actually happen depends on the corpus upon which the system is trained. If “big” occurs much more frequently than “bag”, the sentential context may not be enough to move “bag” ahead of “big”. Frequent words are identified with more efficiency than infrequent words (Monsell, 1991), and so a preference for high frequency words should be a feature of DSHM.

### MALTA

As stated above, ACT-R is a system for modeling human performance of specific tasks. The chunk represents roughly what can be entertained in working memory at one time. The declarative memory system (DM) of ACT-R, in our opinion, is a good model of a task specific short term store. That is, the collection of memories and newly discovered information relevant to the particular task at hand. What is needed to make ACT-R capable of modeling human performance across multiple tasks is a longer term store of persistent knowledge. It is our belief that DSHM can be integrated with ACT-R, so as to provide this persistent knowledge store. There are many possible ways to do this. Two, MALTA1 and MALTA2, will be described here.

The Multiple and Long-term Task Architecture (MALTA), is the name used to reference both the theory underlying MALTA1 and MALTA2, and the implementational features they share in common. MALTA is designed to model the manner in which human beings: transition from completed tasks to new tasks; switch back and forth

between concurrent tasks; and, apply knowledge in a context specific manner to each task. MALTA1 is the most basic implementation of MALTA. It makes minimal changes to how ACT-R operates, and is best suited for simply accounting for how humans transition from completed tasks to new tasks. MALTA2 is more dynamic and involves a more drastic departure from orthodox ACT-R. It is suitable for accounting for how humans switch back and forth between concurrent tasks. In the descriptions of MALTA below, some details are omitted for clarity. All information should be assumed to be represented as DSHM objects, unless otherwise stated.

MALTA consists primarily of a DSHM, a dynamical holographic store of all persistent knowledge available to the system; a set of ACT-R model templates, each with an associated set of *template keys*, represented as DSHM objects; a sensory system (which can be implemented as ACT-R perceptual buffers); and, an executive responsible for integrating the DSHM and the ACT-R portion of the system. Each model template is identical to a standard ACT-R model except that its DM contains only a minimal set of predefined task specific chunks. However, the template may also define additional chunk types that can be used by the executive to properly format objects retrieved by the DSHM, for use by the model, as explained below.

### *MALTA1*

The most basic way to integrate DSHM and ACT-R is to have them operate as independently as possible. MALTA1 is based on this premise. When MALTA1 is first started, and after it completes a task, the executive samples the sensory system and binds this to an optional trace left behind by the previously completed task (if any), to create a

representation of the state of the system. This *state cue* is used to determine which of the ACT-R templates, contained within the given MALTA model, best matches the current context of the system's environment. This is accomplished by determining which set of template keys best match the state cue. The winning ACT-R template is used to invoke an ACT-R model suitable for performing a task relevant to the current environment. The default task specific chunks provided by the template are supplemented by the DSHM. This is accomplished, as follows. Each template specifies some number of chunk definitions. These definitions are bound to the state cue to create a set of context specific incomplete chunks, which are then presented to the DSHM. The DSHM completes these chunks, which are then imported in to the DM of the current ACT-R model.

The ACT-R model created using the template then runs without further interference from the rest of the system. However, as the model runs, all new and modified chunks are tagged with an updated state cue, and fed back into the DSHM. This way, all information acquired while performing a task is made available in the future to the system. When the task is completed, the executive resamples the environment and prompts the application of a new template as described above. Thus, MALTA1 operates as continuous series of ACT-R models, which each makes use of information in the persistent knowledge store (the DSHM) relevant to each task, while committing new information to memory.

### *MALTA2*

MALTA2 operates in the same manner as MALTA1, but with a few differences. At any given moment in time, MALTA2 is both running an ACT-R model based on one

of its templates, and predicting the future state of the system's environment. If that future state is inconsistent with the current task, it abandons the currently running model, and invokes a new one more relevant to what the system anticipates. The executive accomplishes this by first encoding the current state of the environment every time a production fires in the current ACT-R model. This state cue is then bound to representations of the previous 2 states of the system to form a representation of the recent history of the system. The result of this calculation, the *history cue*, [current-state:previous-state:two-states-ago], is first committed to the DSHM. Next, for each chunk in DM, the chunk is converted to a DSHM object, bound to both the current state cue and the current template keys, and then committed to the DSHM. Once this is done, the current state cue is used to predict the next state of the system. This accomplished by creating an incomplete history cue, using the current state and the previous state as cues: [?:current-state:previous-state]. This history cue is completed by the DSHM. The *predicted state cue* used to complete the incomplete history cue is a state cue previously committed to the DSHM, and which in the past typically followed states similar to the current state cue and the previous state cue. See West and Lebiere (2001), for a discussion of this predictive technique, and how it is used to discover sequential dependencies in the evolution of dynamical systems.

The predicted state cue is then compared to the current template keys. If they fail to agree, the current task is judged to be inappropriate for what the system anticipates to be the relevant context of the environment. The current ACT-R model is abandoned, and the predicted state cue is used to invoke a new ACT-R model in the manner that MALTA1 does after the completion of a task. This scheme of tightly binding perceptions

of the environment to cognition is inspired, in part by Lawrence Barsalou's situated simulation theory (2003).

Perhaps the most powerful feature of MALTA is its ability to halt a current task, begin a new one, and then return to its previous task. A task can be resumed by the executive, if the state of the system causes it to invoke a model based on the same template as had previously been in operation. Since the state of the system determines what chunks will be retrieved from DSHM, a previous task can be considered to be continued if it just so happens that the DM system is restored to the state it was in prior to being interrupted. This would be the case if the state of the system returns to (at least nearly) the exact state it was in before the interruption (i.e., the same model template and perception of environment). For example, imagine that the system has invoked a mathematical problem solving model. The system progresses by detecting unsolved problems via the perceptual system, and going about taking the steps to solve them. Suddenly, a spider is detected in the environment of the model. This is inconsistent with doing math, and the executive abandons the mathematical problem solving model, in favour of the 'kill bug' model. Once the spider has been disposed of, the perceptual system returns to detecting unsolved math problems, and the mathematical problem solving model is reinvoked. Chunks pertaining to partially completed problems will be restored if the system detects a problem that it had been previously working on.

### *Resolving the Issues*

We believe that the features of MALTA described above will resolve some of the limitations of ACT-R, with respect to retaining task neutral persistent knowledge. The

reason for this is that DSHM operates differently than DM. ACT-R creates, modifies, and stores explicit representations of chunks in DM, in the context of a specific task. However, it is not clear that the manner in which a chunk is structured during one task will make it suitable for retrieval during a different task. A set of special productions could potentially be designed to convert one set of chunks to another. However, this seems to be a redundant process at a minimum, and most likely a very difficult one to implement efficiently in a methodologically sound manner.

DSHM does not store an explicit representation of an object when it is imported into memory, as is the case with chunks and DM. Rather, DSHM stores, implicitly in the memory vectors of each component object of a complex object, the relations that exist between those components. As a result, every object in the DSHM stores information about every context in which it has existed, due to the binding mechanisms described above. Thus, each object can be associated with potentially thousands of other objects without any of these associations being explicitly encoded anywhere in the system. These associations can reflect very complex relations pertaining to the individual contribution of each object to the structures of other objects. That is, every object 'knows' about the kinds of structures to which it contributes, and what other objects were part of those structures. We believe these features to be an advantage over models of long-term memory that rely on explicitly encoded connections between entities in memory (e.g., Schultheis, Barkowsky, & Bertel, 2006).

A central feature of MALTA is that it is not limited to retrieving only exact reproductions of previously stored chunks from DSHM. Just as synonyms group together in BEAGLE, so do both structurally similar complex objects and conceptually similar

atomic objects group in DSHM. Each stored chunk can be interpreted as making more salient an attractor in the dynamical structural/conceptual state-space formed by DSHM; see van Gelder & Port (1995) for a review of dynamical systems accounts of cognition. A retrieval cue consists of two parts: the structure of an incomplete object, and the information contained in the complete components of that object. The influence of the complete components is roughly to pick out a point in the state-space of the DSHM. The objects located around this point are evaluated for their suitability as candidates for filling in the missing components of the incomplete object. Thus, novel chunks can be generated when the converging contributions of the supplied retrieval cues pick out objects from DSHM that have not previously been brought together according to the constraints placed on that retrieval.

This feature of DSHM allows for chunks, specific to a particular context, to be generated without that exact context having been presented to the given MALTA model previously. This ability bears a certain similarity to analogical reasoning, the importance of which, in understanding cognition is not to be underestimated (Gentner, Holyoak, & Kokinov, 2001). Thus, the potential for MALTA to enjoy this ability will not be overstated here. However, in developing MALTA we plan to leverage this ability as much as possible.

#### References

- Anderson, J. R., & Lebiere, C. (Eds.) (1998). *The atomic components of thought*. Mahwah, NJ: Lawrence Erlbaum.

- Atkinson, R. C., & Shiffrin, R. M. (1968). Human memory: A proposed system and its control processes. In K. W. Spence and J. T. Spence (Eds.), *The Psychology of learning and motivation: Vol 2. advances in research and theory*, (pp. 89-195). New York: Academic Press.
- Baddeley, A. D. & Hitch, G. (1974). Working memory. In G. H. Bower (ed.) *Recent advances in learning and motivation: Vol. 8.* (pp. 47-90). New York: Academic Press.
- Barsalou, L. W. (2003). Situated simulation in the human conceptual system. *Language and Cognitive Processes*, 18, 513–562.
- Byrne, M. D., & Anderson, J. R. (1998). Perception and action. In J. R. Anderson & C. Lebiere (Eds.), *The atomic components of thought*, (pp. 167-200). Mahwah, NJ: Lawrence Erlbaum.
- Craik, F. I. M. & Lockhart, R. S. (1972). Levels of processing: A framework for memory research. *Journal of Verbal Learning and Verbal Behavior*, 11, 671-684.
- Gentner, D., Holyoak, K. J., & Kokinov, N. (Eds.) (2001). *The analogical mind: Perspectives from cognitive science*. Cambridge, MA: MIT Press.
- Jones, M. N., & Mewhort, D. J. K. (2007). Representing word meaning and Order information in a composite holographic lexicon. *Psychological Review*, 114, 1-37.
- Kieras, D. E., & Meyer, D. E. (1997). An Overview of the EPIC architecture for cognition and performance with application to human-computer interaction. *Human-Computer Interaction*, 12, 391-438.
- Laird, J., Newell, A., & Rosenbloom, P. S. (1987). Soar: An architecture for general intelligence. *Artificial Intelligence*, 33, 1-64.

- Mauro, D. G., LeFevre, J., & Morris, J. (2003). Effects of problem format on division and multiplication performance: Division facts are mediated via multiplication-based representations. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, 29, 163–170.
- Monsell, S. (1991). The nature and locus of word frequency effects in reading. In D. Besner & G. W. Humphreys (Eds.), *Basic processes in reading: Visual word Recognition*. (pp. 148–197). Hillsdale, NJ: Lawrence Erlbaum.
- Newell, A. (1990). *Unified theories of cognition*. Cambridge, MA: Harvard University Press.
- Plate, T. A. (1995). Holographic reduced representations. *IEEE Transactions on Neural Networks*, 6, 623-641.
- Pribram, K. H. (1971). *Languages of the brain: Experimental paradoxes and principles in neuropsychology*. Englewood Cliffs, NJ: Prentice-Hall.
- Pribram, K. H. (1991). *Brain and perception: Holonomy and structure in figural processing*. Hillsdale, NJ: Lawrence Erlbaum.
- Rosch, E. H. (1975). Cognitive representations of semantic categories. *Journal of Experimental Psychology: General*, 104(3), 192-233.
- Rutledge-Taylor, M. F. (2005). Can ACT-R realize “Newell’s Dream”? In B. G. Bara, L. Barsalou & M. Bucciarelli (Eds.) *Proceedings of the 27<sup>th</sup> Annual Conference of the Cognitive Science Society* (pp.1895-1900). Stresa, Italy: University and Polytechnic of Turin.
- Samuel, A. G. (1981). Phonemic restoration: Insights from a new methodology. *Journal of Experimental Psychology: General*, 110, 474-494.

- Schultheis, H., Barkowsky, T., & Bertel, S. (2006). LTM<sup>C</sup> — An improved long-term memory for cognitive architectures. In D. Fum, F. D. Missier, & A. Stocco (Eds.) *Proceedings of the Seventh International Conference on Cognitive Modeling* (pp. 274-279). Trieste, Italy
- Tulving, E. (1972). Episodic and semantic memory. In E. Tulving and W. Donaldson (Eds.) *Organisation of memory*. London: Academic Press.
- Van Gelder, T. J., & Port, R. (1995). It's about time: An overview of the dynamical approach to cognition. In R. Port & T. van Gelder (Eds.), *Mind as motion: Explorations in the dynamics of cognition*. (pp. 1-43). Cambridge, MA: MIT Press.
- West, R. L., & Lebiere, C. (2001). Simple games as dynamic, coupled systems: Randomness and other emergent properties. *Cognitive Systems Research*, 1(4), 221-239.

# MODELING THE FAN EFFECT USING DYNAMICALLY STRUCTURED HOLOGRAPHIC MEMORY

Previously published as Rutledge-Taylor, M. F. & West, R. L. (2008). Modeling The fan effect using dynamically structured holographic memory. In B. C. Love, K. McRae, & V. M. Sloutsky (Eds.), *Proceedings of the 30<sup>th</sup> Annual Conference of the Cognitive Science Society* (pp. 385-390). Washington, DC: Cognitive Science Society.

## Contribution

This paper supports the practical component of the dissertation by providing an example of a basic cognitive phenomenon, the fan effect, that DSHM is able to model. The fan effect provides a good test bed for DSHM as it is a well understood memory phenomenon and has been modeled using ACT-R (Anderson & Reder, 1999; West, Pyke, Rutledge-Taylor &, 2010).

Important interpretations of DSHM theory include the use of DSHM confidence values as the basis for reaction time predictions, as well as the use of item completion to produce multiple answers to recall questions.

## Abstract

Dynamically Structured Holographic Memory (DSHM), a system based on holographic reduced representations, designed for modeling memory is described. A DSHM model of the fan effect (Anderson, 1974) is presented. Comparisons between the DSHM model and an existing ACT-R model (Anderson & Reder, 1999) are made. It is

argued that DSHM can be interpreted as providing a lower-level account of several features of ACT-R such as activation, noise, and the ability to make errors.

### Introduction

The aim of this paper is to demonstrate that DSHM embodies features of human memory, and does so in a way that is emergent from the manner in which information is structured within it. As such, DSHM provides an account of the implementation of several higher-level features of memory, such as activation and noise, which architectures such as ACT-R take to be basic. The value of this account is that it may help to bridge symbolic account of cognition (i.e., ACT-R), with neurally inspired accounts, especially those that make use of vector representations, e.g., the Neural Engineering Framework (Eliasmith & Anderson, 2003), and holographic reduced representations (Stewart & Eliasmith, 2008).

### The Fan Effect

The fan effect is a well known memory phenomenon relating to the time required to affirm (or, reject) a given proposition as true. The effect was first described by J. R. Anderson (1974). Anderson generated a study set of 26 sentences each of which described a fact about one of 16 persons residing in one of 16 places; the phrase ‘the hippie is in the park’ is a well known example. The fan of a word is the number of sentences from the study set in which it appears. The fan of each person and place in the study set is one, two or three. Thus, the 26 sentences can be divided into cells of a three by three table according to the fans of the two content words. Anderson had each

experimental participant memorize the study set, and then measured the time that elapsed between the subsequent presentation of a sentence and the participant's signal that the sentence was either a member, or not a member of the study set. The results showed that there was a positive correlation between reaction time and the fan of the sentence (the sum of the fans of each of the content words in a test sentence). For example, the time to judge a sentence appearing in the study set with fans of one for both the person and place was 1178 milliseconds, while for a sentence with fans of three for both words was 1514 milliseconds.

As a basic phenomenon of human memory, the fan effect has served as a useful paradigm for testing theories of memory retrieval, including those pertaining to recall time and accuracy. DSHM is primarily a theory of how information is represented and structured in memory. However, as such, it also offers mechanisms for retrieving information from memory. Therefore, it ought to be consistent with the prevailing theories of memory retrieval. Mechanisms such as spreading activation (Anderson & Reder, 1999), processes such as retrieval effects (Anderson & Reder, 1999), representation effects (Goetz & Walters, 2000; Radvansky, et al., 1993), and memory capacity (Bunting et al., 2004) have all been proposed as responsible for, or playing a role in, producing the fan effect. The basic mechanisms in DSHM are cue based retrieval and interference. However, informational structures extracted from DSHM can be interpreted as having a particular activation value and are subject to spreading activation

## DSHM

DSHM is a model of memory that makes use of holographic reduced representations (HRR). It is based on Jones and Mewhort's BEAGLE model of the lexicon (Jones & Mewhort, 2007). The details of the DSHM architecture and the similarities between BEAGLE and DSHM can be found elsewhere (Rutledge-Taylor & West, 2007). Here, only the details necessary for understanding the application of DSHM to the fan effect will be included.

Objects in the DSHM system, referred to as items, are represented as pairs of vectors. Every vector in the system is the same length, which typically consists of hundreds or thousand of elements. Each vector consists of real valued elements and has a Euclidean length of 1.0. The *environmental* vector is a unique static internal representation of the item. The *memory* vector is dynamic and stores all of the associations existing between the given item and every other item in the system, from the given item's perspective. An item may also reference a set of sub-items, from which it is composed. This set of sub-items is flagged as bearing either an ordered or unordered relationship to one another, (e.g., the words in a sentence where order matters versus the keywords of a journal article where the order of the words does not). This recursive structure of items allows any form of information that can be represented as a tree to be represented as a DSHM item. Items without sub-items are referred to as terminal or atomic items.

A DSHM system defines a multi-dimensional state-space, where each environmental and memory vector is a point on the surface of a hypersphere (with radius 1.0). In such a space the cosine of any two vectors is a measure of their distance, where a

value of 1.0 indicates that the vectors are identical, and 0.0 indicates that they are orthogonal. The cosine of any two memory vectors is roughly a measure of the similarity of the items. Specifically, two items that can be exchanged for one another in many different informational patterns (complex items) will have common neighbors which will affect their memory vectors in similar ways. The cosine of the environmental vector of one item, properly encoded, and the memory vector of another item is a measure of the items' co-occurrence frequency. That is, they occur together in the same informational patterns. Here is an analogy: in a DSHM model of bus drivers and passengers, two bus drivers that drive the same route have *similar* knowledge (memory vector) because they experience the same traffic and interact with the same sets of regular passengers. As such, they are *interchangeable*. A driver and a regular daily passenger are not interchangeable. However, they have knowledge of one another's daily routines because they co-occur together on the bus. Their respective environmental vectors influence each other's memory vectors.

### *Encoding*

Information is entered into a DSHM system by presenting it with a complex object representing that information. When this happens the object is converted into a DSHM item. As a reminder to the reader, think of a DSHM item as a tree, where the head of each non-terminal sub-tree is a complex item itself. Once represented as a complex item, hereafter the aggregate, every item that exists as a component of the aggregate is informed that it is associated with every other item in the aggregate. This is done by binding the environmental vectors of various substructures of the aggregate

together and adding the results to the memory vectors of the other items in the aggregate. The binding of vectors is achieved via circular convolution (Plate, 1995). The process makes the memory vector of each item in the aggregate pattern more similar to the environmental vectors of each other item than they were previously.

For example, a proposition such as the 'the hippie is in the park' can be represented as an item consisting of only the content words, [hippie:park]. When entered into the system, this co-occurrence of the items causes the memory vector of the item representing 'hippie' to become more similar (i.e., closer in the multidimensional space) the environmental vector of the item representing 'park', and vice-versa (ignoring some computational details).

### *Decoding*

Information is retrieved from DSHM by presenting the system with incomplete patterns. These patterns take the shape of DSHM items. The components of the pattern that are complete (i.e., not missing) are used to generate a set of probe vectors that point to locations in the state space defined by the system. The proximity of a probe and the environmental vector of an item in the system, quantified by calculating the cosine of the two vectors, is a measure of how well the item matches the probe. Thus, given a probe cue, every item in the system can be assigned a value which is relative to the cue. The cumulative influence of each of the probes generated is used to produce a rank ordered list of candidate completions of the incomplete pattern presented to the DSHM system, each with an associated match value.

### *Cognitive Interpretation*

Traditionally, memory retrieval can be divided into two types: recall and recognition. In general terms, the former can be understood as retrieving some content from memory that matches some cue, such as the content provided in a question like ‘where are the hippies?’. Here, the words ‘where’ and ‘hippie’ cue the recall of places known to be populated by hippies. The latter can be understood as affirming that a presented fact or item is known, true, or has been studied. Asking a participant whether he or she has studied the proposition ‘the hippie is in the park’ is an example of a recognition memory task. In this case, the proposition as a whole and each content word in the proposition is a potential cue to memory. Participants of experiment 1 of Anderson (1974) were tasked with both recall and recognition, the former during the *learning* phase, and the latter during the *reaction-time* phase.

A model of memory must explain the mechanisms that underlie both recall and recognition. These mechanisms must account for how these processes can both succeed and fail. In the case of recall, success is a matter of only retrieving correct answers to questions, or otherwise providing only the appropriate matches to the provided cues. Failure is to either omit correct responses, or retrieve incorrect responses. Successful recognition is a matter of affirming true facts and correctly rejecting false ones, while failure is to deny true facts or to affirm false ones.

### *Where are the Hippies?*

During the learning phase of Anderson’s fan effect experiment, participants were first asked to rehearse the study set of sentences. To confirm that they could remember

each of the sentences, the participants were asked questions of the form “who is in the park?” and “where are the hippies?” (Anderson, 1974). For each question there is exactly one, two or three correct answers, depending on the fan of the content word supplied in the question. A challenge to any model of memory is to explain how all and only the correct answers to a question are recalled.

To ask the DSHM system to answer a recall question an incomplete pattern encoding the question is presented to it. For example, to ask the system ‘where are the hippies?’, the incomplete pattern [hippie:?] is presented to it. As discussed above, every item in the system is assigned a value, corresponding to how well it matches the cue (in this case ‘hippie’). As a matter of how the state-space of the system organizes itself, only items that have co-occurred with the cue will be activated to a significant extent. Thus a distinctive drop-off in activation values between those items associated with the probe and those that are not can be observed. An important assumption that we have made is that the items before the drop are recalled and the items after the drop are not.

Table 11 displays the top six answers to the questions: “where are the hippies?”, and “where are the lawyers?”, presented to a DSHM system with vector length of 2048, trained only on the study set. The word ‘hippie’ has a fan of one, while ‘lawyer’ has a fan of three. Therefore, there is only one correct answer to former question, and three correct answers to the latter. The system correctly scores only one answer to the first question highly, and three to the second. This precise response profile corresponds to the behaviour of an experimental participant with extremely good memory. Errors can be produced by a model with a smaller vector length, corresponding to a participant with

poorer memory. A small vector length limits the memory capacity of the model, and makes it more vulnerable errors caused by memory interference.

*Table 11.* Where are the hippies?

Where are the hippies?		Where are the lawyers?	
Answer	Value	Answer	Value
[hippie:park]	0.041382	[lawyer:store]	0.013675
[hippie:store]	0.000007	[lawyer:park]	0.010719
[hippie:captain]	0.000003	[lawyer:bank]	0.008042
[hippie:cave]	0.000001	[lawyer: fireman]	0.00001
[hippie:debutante]	0.000001	[lawyer:cave]	0.000003
[hippie:church]	0.000001	[lawyer: captain]	0.000002

The drop-off in activation values serves as a mechanism for determining which items the system will recall, and which it will not. This makes intuitive sense. When a human participant is asked “where are the lawyers?”, he or she will not answer with a rank ordered list of all 16 places mentioned in the study set. Rather, he or she will respond with only a few answers that seem more correct than the other possible answers. The ability of DSHM to provide several appropriate answers to a single question, when more than one answer is correct, is one of the strengths of the system.

*Is the Hippie in the Park?*

During the test phase of the experiment, a set of test sentences is presented to the participant, one sentence at a time. The task of the participant is to decide as quickly as possible, while maintaining a high degree of accuracy, whether the presented sentence was a member of the study set or not (Anderson, 1974). The DSHM model makes this membership judgment not by searching memory for the presented proposition, rather it assesses how tightly associated the content words in the proposition are. This is done by asking itself “where are the hippies?”, and ‘who is in the park?’. The average of the activation values associated with the correct answers (i.e., ‘park’, and ‘hippie’) determines the activation of the proposition as a whole. An incorrect rejection of a study set sentence occurs, when the model fails to recall ‘park’ as an answer to the former question, and vice-versa. This is possible due to the cut-off mechanism described above.

*Spreading Activation*

Pattern completion can be interpreted as a form of spreading activation in which activation spreads out from the probe and weakens as it spans the state-space (i.e., the closer to the probe, the higher the activation). Thus the match value would correspond to a cue specific activation level. We interpret activation level as determining how much time it takes for each item to be retrieved (in a manner similar to ACT-R).

When a proposition is recognized it can be affirmed instantly, without delay. However, in the case of foils, the model does not know in advance how many items will be recalled by the ‘who’ and ‘where’ sub-questions. Therefore, there may be a hesitation associated with waiting to make sure that no more items (possibly including a match) are

forthcoming. For example, if presented with the foil “the hippie is in the store”, the sub-question “where are the hippies?” will quickly cause the recall of ‘park’. However, the fact that ‘store’ will not make the cut-off cannot be verified until all of the activated items have been recalled. Thus, a brief period when no more items are recalled must follow the last recalled item to ensure that the cut-off has been reached. Here, an analogy to cooking microwave popcorn can be made. When cooking microwave popcorn, the instructions typically state that the popcorn bag is to remain in the oven while the gaps between pops do not exceed a few seconds. The need to keep the bag in the oven can be verified the instant a pop is heard. However, to reject this need and remove the bag one needs to wait a few seconds after hearing the latest pop to verify that the maximum distance between pops has elapsed. By analogy, to reject a foil also requires a hesitation to confirm that the target proposition is not about to ‘pop’ into mind. If ‘store’ had been recalled, (or ‘hippie’ in the case of the “who is in the store?” sub-question), an incorrect affirmation of the foil would occur.

#### DSHM Model of the Fan Effect

The DSHM model of the fan effect follows the original conditions specified in experiment 1 of Anderson (1974). The sentences were represented as complex items composed of an ordered list of two items representing a person and place, e.g., [hippie:park]. The omission of function words in the representation of the sentence does not significantly affect the important features of the memories for these sentences; they are redundant to the information contained in the content words, in this case.

Special care was taken to ensure that the DSHM participants followed as closely as possible the exact procedures as were followed by the human experimental participants. The DSHM models: 1) rehearsed the study set; 2) confirmed knowledge of the study set by answering recall questions of the form ‘where are the hippies?’; and, 3) answered recognition questions of the form, “was ‘the hippie is in the park’ a member of the study set’. When errors were made during confirmation, by responding with too few correct answers (e.g., ‘the lawyer is in the store’, and ‘the lawyer is in the park’, but omitting ‘the lawyer is in the bank’), or with at least one incorrect answer (e.g., the correct answer, ‘the hippie is in the park’, plus the incorrect answer, ‘the hippie is in the store’), the models were made to rehearse the propositions for which recall errors were made, and retested, just as the human experimental participants were.

A parameter we chose to manipulate is how much background knowledge each model instance has preloaded before learning the study set. Background knowledge was represented as a variable amount of [person:random number], and [random number:place] items (e.g., 'hippie:7845'). The random numbers represent arbitrary facts the participant may know about the places and persons in advance of the experiment; e.g., the participant may know that hippies live in an apartment in his or her building, or that there are pigeons in the park, etcetera. We hypothesized that background knowledge could affect a model’s ability to distinguish between sentences from the study set and foils by increasing the number of facts that may interfere with one another in the system. It also adds an element of ecological validity to the simulations in that no human participant enters into the experimental scenario without some knowledge relating to the persons and places, as concepts, appearing in the study set.

### *Generating Reaction Times*

Reaction times are derived from the activation values of the sentences produced by the system. Thus, to convert activation to reaction time, some theory of this relationship must exist. At present we have not developed a universal function that relates activation to reaction time. Here we will use an ad hoc formulae so as to make comparisons between the DSHM models, the human data from Anderson (1974) possible. For affirmations we used the formula:  $670\text{ms} + 70\text{ms} * 1/a$ ; where  $a$  is the activation of the affirmed proposition. For rejections we used the formula:  $770\text{ms} + 70\text{ms} * 1/r$ ; where  $r$  is the activation of rejected proposition. The additional 100 milliseconds added to the reaction times for rejections is intended to compensate for characteristic delay associated with judging foils, discussed above.

### *Generating Errors*

Human participants incorrectly judged the study set membership of a test sentence 3.9% of the time in experiment 1 of Anderson (1974). Thus, an adequate model of the fan effect should make a similar number of errors. Since errors in DSHM systems only occur when the memory capacity of the system is saturated, a range of vector lengths were tested to determine which makes errors with a frequency approximating that of human participants.

### *The Simulations*

The two parameters discussed above were manipulated so as to produce several populations of models, each with common parameters. Vector length values in the set {32, 128, 512, 2048} and background knowledge parameters in the set {0, 1, 2} were

used. The background knowledge parameter is equal to the number to additional random propositions the model is trained on for each of the 16 persons and 16 places; e.g., for a value of 1, one random fact of the form [person:random number], for every person, and similarly for every place, was entered into the model's memory prior to learning the study set. In the case of no background knowledge, the study set was initially read once during the study phase, and twice otherwise. This ensured that every sentence in the study set was read at least one time more than any of the background knowledge sentences.

The combinations of these parameters make 12 unique sets of simulations. There were 100 simulated participants for each set.

### *Results*

For each of the 12 sets of simulations the mean reaction times and error rates for each of the nine kinds of sentences, differentiated by the fans of the content words were recorded. Recall, the person and place appearing in each sentence has a fan of one, two or three; this makes nine combinations. Data was collected for both true sentences, those appearing in the study set, and for false sentences, or foils, those not appearing in the study set.

There were several general patterns that emerged from these simulations: 1) mean reaction time was somewhat larger for the lower vector lengths as compared to the larger vector lengths; 2) the standard deviation of the reaction times was dramatically larger for the lower vector lengths as compared to the larger vector lengths; 3) judgment errors were similar to human performance for only the sets with a vector length of 128; 4) although the amount of background knowledge did not significantly affect mean reaction

time, it did increase the standard deviation of the reaction times for all vector lengths, except for the sets with a vector length of 32; 5) the behavior profile of the models was approximately equal for both true sentences and foils.

Figure 25 illustrates points 1 and 2 above. In order to simplify the figure, only the true sentences with fans of two and six are depicted. The fan of a sentence is the sum of the fan of the person and place. These values illustrate the fan effect pattern in the data, for each vector length. The bars indicate the standard deviations on these mean reaction times in milliseconds. The figure truncates the standard deviation of 3671 milliseconds for the simulations with vector length of 32 and fan of six. This decrease in the fan effect with vector length is consistent with studies showing a decrease in the fan effect with working memory capacity (Bunting et al., 2004).

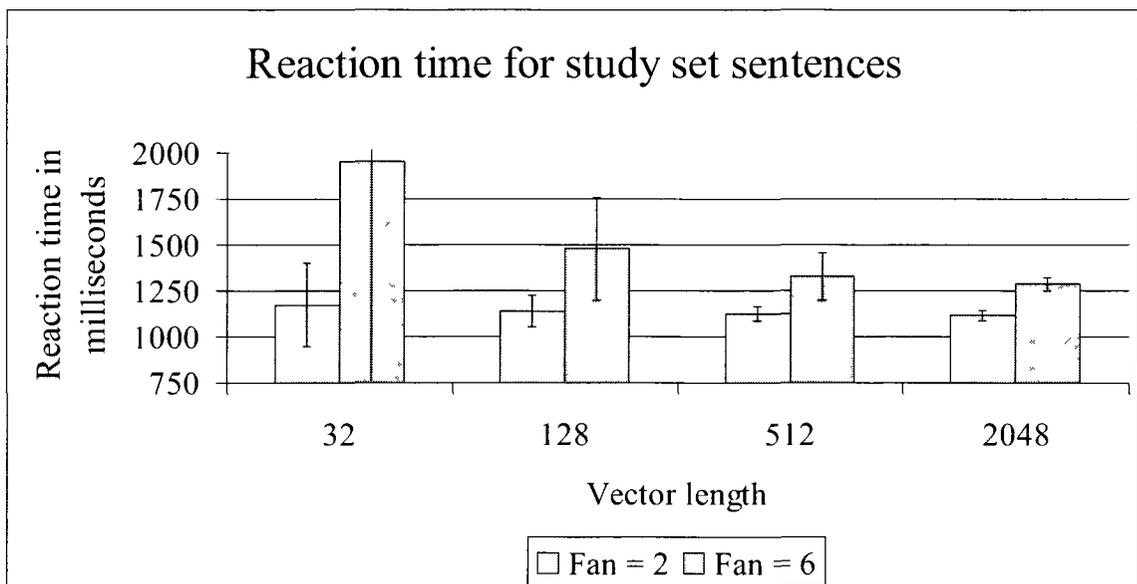


Figure 25. Reaction time for study set sentences.

Figure 26, including human data from Anderson (1974) illustrates point 3. The set of models that best matched the human data belonged to the set with vector lengths of 128. They were the only models that produced an error rate resembling human performance. Additionally, and informally, the standard deviation of the reaction times for these models resembles a ‘human-like’ profile, unlike for the other vector lengths. Unfortunately, Anderson (1974) did not include standard deviations on his human data, and so we could not compare the human data and the model on this measure.

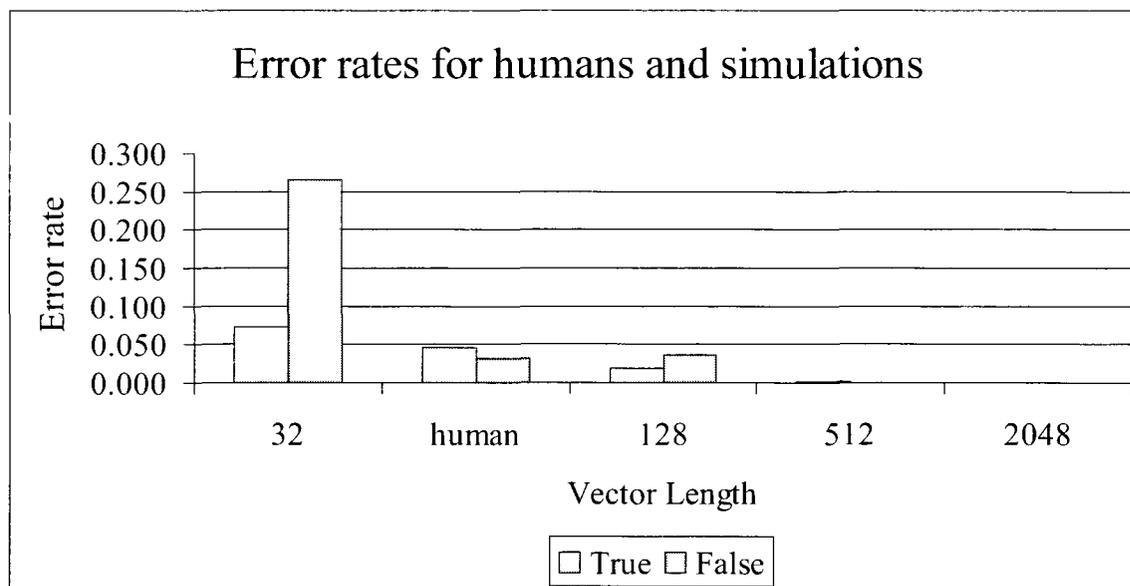


Figure 26. Error rates for humans and simulations.

Tables 12 and 13 present the mean reaction time and error rate data for the models with a vector length of 128, in the same format as the human data is presented in Anderson (1974). The rows are organized by the fan of the place, while the columns are organized by the fan of the person. Each cell contains a mean reaction time in

milliseconds and an error rate for propositions with the fans corresponding to the given cell. As with Anderson (1974), only the times corresponding to the affirmation of true sentences and the rejection of false sentences were included in the reaction times; i.e., the times associated with errors (misses and false positives) were not included.

*Table 12.* Model reaction times and error rates for trues

Place fan	Person fan			Mean
	1	2	3	
1	1137	1191	1240	1189
	0.000	0.002	0.000	0.001
2	1166	1285	1377	1276
	0.002	0.027	0.038	0.022
3	1232	1371	1478	1360
	0.000	0.032	0.072	0.034
Mean	1178	1282	1365	1275
	0.001	0.020	0.037	0.019

#### Evaluation of the Model

The reaction times for the DSHM model were generated according to ad hoc functions designed to produce values that match the human data, on average. Thus, gross agreement between the DSHM reaction times and the human reaction times is a given. However, the functions are linear and do not finesse the data so as to produce the fan and

min effects (Anderson, 1974). Thus, a comparison between the DSHM model and the human data from Anderson (1974) is provided. Figures 27 and 28 illustrate the close match in the pattern of the reaction times, for sentences with fans of two, three, four, five, and six for true sentences and false sentences respectively. Also, it should be noted that the error bars represent predictions of what the level of variability in the human results should be, but this could not be confirmed as that data was not available.

*Table 13.* Model reaction times and error rates for fables

Place fan	Person fan			Mean
	1	2	3	
1	1170	1225	1320	1238
	0.003	0.02	0.042	0.022
2	1242	1288	1437	1322
	0.018	0.023	0.048	0.03
3	1323	1413	1488	1408
	0.045	0.057	0.072	0.058
Mean	1245	1308	1415	1323
	0.022	0.033	0.054	0.036

#### Relationship to the ACT-R Model

Although relying on very different mechanisms, the DSHM model is consistent with the ACT-R model of the fan effect (Anderson & Reder, 1999). For both models, the

characteristic pattern of reaction times defining the fan effect results from patterns in the activation levels of the propositions learned by the systems.

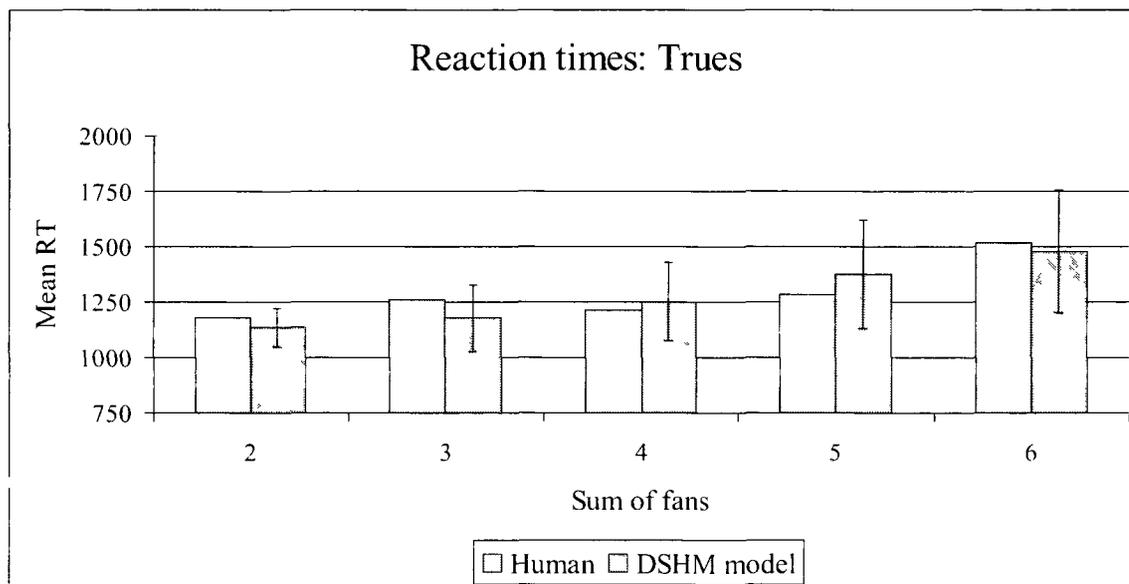


Figure 27. Reactions times for study set sentences. Bars indicate standard deviation.

The activation of a target sentence in DSHM is a function of the association between the person and the place appearing in the sentence. Mathematically, this is a measure of the distance between the environmental vector of each word, properly encoded, and the memory vector of the other word in the state-space defined by the system.

The activation of a target sentence according to the ACT-R model is a function of the associations between the chunk representing the target sentence and the chunks representing each of the person and place. Mathematically, it is equal to the sum of a base level of activation and a weighted sum of the activations of the person and place. In

the case of foils, the target sentence does not exist in the declarative memory system of the model and cannot be retrieved. Instead, a chunk, partially matching the target, which contains one of the two content words is retrieved. As a result of the retrieved chunk and the target sentence chunk having only one of the person and place in common the net activation of the retrieved chunk has one less source of activation, and thus, on average, has a lower net activation value than the average true sentence.

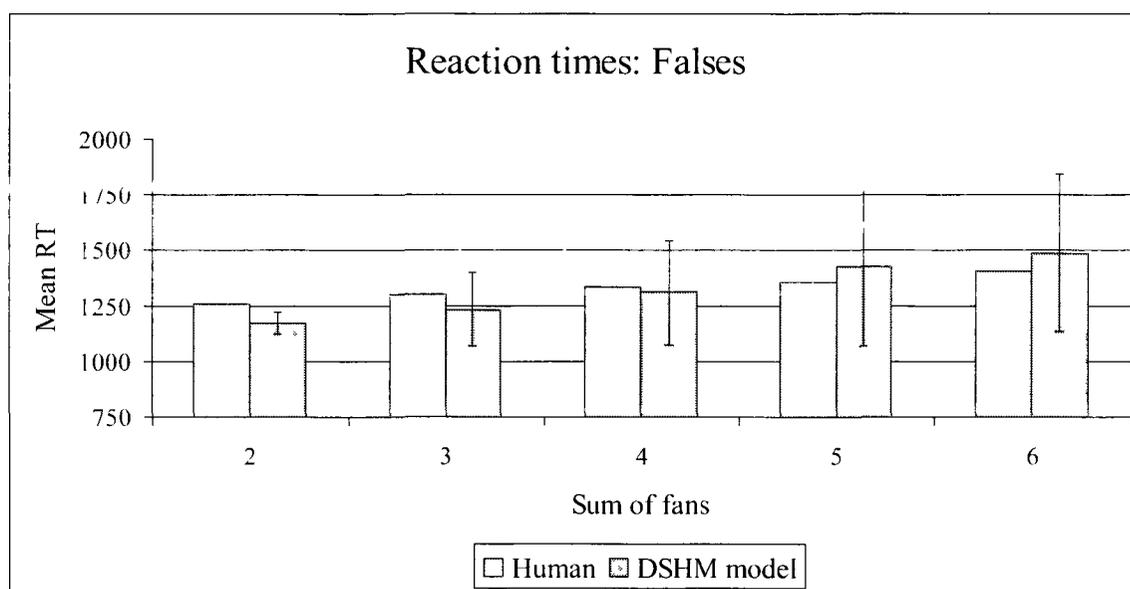


Figure 28. Reaction times for foils. Bars indicate standard deviation.

Anderson & Reder (1999) do not discuss how errors could be produced by the ACT-R model. Presumably, errors could be caused by increasing the amount of noise in the system, and/or raising the activation threshold for recall.

We are of the belief that the DSHM system and the ACT-R architecture are compatible. The DSHM and ACT-R models of the fan effect both agree with the human

data. This agreement is a result of DSHM and the declarative memory system of ACT-R providing alternative accounts of the same psychological memory mechanisms. We claim that ACT-R provides a higher-level, more abstract, account of memory, making use of mathematical formula to describe the activations of chunks in memory. DSHM, on the other hand, produces the same profile of behavior as an emergent property of the distributed representation of associations between items in the state-space of the system. For example, ACT-R activation can be interpreted as emergent on DSHM association strength, and ACT-R noise as emergent on DSHM vector length (memory capacity). As such DSHM resides hierarchically between ACT-R as an abstract computational model of memory and lower-level neural network models of memory, such as those described in Goetz and Walters (2000).

### Discussion

We have shown that a holographic model of memory can account for the fan effect as an emergent property of the manner in which information is represented in the system. In addition to predicting reaction times it also predicts errors of omission and commission. A strength of the system is that it can also model the training portion of the fan effect experiment, which involves the ability to recall multiple correct answers to questions. Finally, the fan effect was produced by training and testing the model in exactly the same manner as were the human participants, which jointly validates both the encoding and retrieval aspects of the model.

## References

- Anderson, J. R. (1974). Retrieval of propositional information from long-term memory. *Cognitive Psychology*, 6, 451-474.
- Anderson, J. R. & Reder, L. R. (1999). The fan effect: New results and new theories. *Journal of Experimental Psychology: General*, 128(2), 186-197.
- Bunting, M. F., Conway A. R. A., & Heitz, R. P. (2004). Individual differences in the fan effect and working memory capacity. *Journal of Memory and Language*, 51(4), 604-622.
- Eliasmith, C., & Anderson, C. H. (2003). *Neural engineering: Computation, representation and dynamics in neurobiological systems*. Cambridge, MA: MIT Press.
- Goetz, P. & Walters, D. (2000). A neuronal basis for the fan effect. *Cognitive Science*, 24(1), 151-167.
- Jones, M. N., & Mewhort, D. J. K. (2007). Representing word meaning and Order information in a composite holographic lexicon. *Psychological Review*, 114, 1-37.
- Plate, T. A. (1995). Holographic reduced representations. *IEEE Transactions on Neural Networks*, 6, 623-641.
- Radvansky, G. A., Spieler, D. H., & Zacks, R. T. (1993). Mental model organization. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, 19, 95-114.
- Rutledge-Taylor, M. F. & West, R. L. (2007). MALTA: Enhancing ACT-R with a holographic persistent knowledge store. In D. S. McNamara & J. G. Trafton

(Eds.), *Proceedings of the 29<sup>th</sup> Annual Conference of the Cognitive Science Society* (pp. 1433-1438). Nashville, TN: Cognitive Science Society.

Stewart, T. C. & Eliasmith, C. (2008). Building production systems with realistic spiking neurons. In B. C. Love, K. McRae, & V. M. Sloutsky (Eds.), *Proceedings of the 30<sup>th</sup> Annual Conference of the Cognitive Science Society* (pp. 1759-1764). Washington, DC.

## MODELING A THREE TERM FAN EFFECT

Previously published as Rutledge-Taylor, M. F., Pyke, A. A., West, R. L. & Lang, H. (2010). Modeling a three term fan effect. In D. D. Salvucci & G. Gunzelmann (Eds.), *Proceedings of the Tenth International Conference on Cognitive Modeling* (pp. 211-216). Philadelphia, PA: Drexel University.

### Contribution

This paper provides another example of DSHM ability to account for the fan effect. The model had difficulty matching the human reaction times for foils (sentences not in the study set). This revealed the need to produce a more nuanced account of foils.

### Abstract

A fan effect experiment where participants perform recall and recognition tasks on a study set of sentences with three content words was conducted. The aggregate results confirm a fan effect (Anderson, 1974). A model of the recall and recognition tasks was created using Dynamically Structured Holographic memory (DSHM). A comparison to the human data is presented. A discussion of the current resonance based mechanisms in DSHM for generating recognition accuracy and reaction time data is presented. This is contrasted with a previously employed retrieval based mechanism.

### Introduction

The purpose of this paper is to report the results of a fan effect style experiment and to demonstrate that these results can be captured by Dynamically Structured

Holographic memory (DSHM). The experiment conducted was similar to the classic fan effect paradigm (Anderson, 1974).

In Anderson's original experiment, participants studied a set of sentences that contained two content words: a person and a place (e.g., "the hippie is in the park"). Each content word appeared in one, two, or three different sentences. The number of sentences in which a word appears is the *fan* of that word. Each sentence is assigned a fan, which is the sum of the fans of the content words in the sentence. For example, if 'hippie' appeared in three sentences while 'park' appeared in one sentence, 'hippie' would have a fan of three, 'park' would have a fan of one, and the sentence 'the hippie is in the park' would have a fan of four. The results of a recognition task performed on the sentences (and an equal number of foils) demonstrated that the time required to affirm or reject a sentence as a member of the study set was correlated with the fan of the sentence.

The present work extends prior research on the fan effect, and models thereof. We explore the generality of the fan effect by examining memory performance for sentences with three content terms rather than just two (e.g., Anderson, 1974). Additionally, our sentences had a wider range of fans than have typically been studied (or modeled).

### The Three Term Fan Experiment

#### *Method*

Twenty seven participants (11 males and 16 females: mean age 20.0 years, SD = 2.2) were recruited from introductory psychology courses at Carleton University to take part in the experiment. Participants received course credit for their time. Participants

took part in the experiment one at a time. The experiment was divided into three main phases: A study phase, a recall phase and a recognition phase.

In the study phase each participant was assigned one of three unique sets of study sentences and was instructed to memorize the sentences in the list. Once the participant indicated that he or she was prepared to proceed, the recall portion of the experiment began.

The study set consisted of sixteen sentences of the form, “The *color thing* is in the *place*”. The color term was one of ten colors; the thing was one of ten house-hold items; and the place was one of ten locations in/around a typical home. Very typical item/locations combinations, such as ‘comb’/‘bathroom’, were omitted when generating the study set sentences. Eight terms from each category appeared in one study sentence each, while two terms from each category appeared in four sentences each. No two terms appeared together in more than one sentence. For example, if “The orange comb is in the garage” was a member of the study set, no other sentence in the study set described an orange comb, a different colored comb in the garage, or any other orange object in the garage. However, these combinations could occur in foil sentences.

The fan of a sentence is the sum of the fans of the terms in the sentence. Thus, the four possible sentence fans were: 3, 6 9, and 12. The fan effect predicts that judgments for sentences with higher fans should take longer (i.e., have higher reaction times) than for sentences with lower fans. Additionally, the truth of sentences with a higher fan should be recognized with less accuracy than sentences with a lower fan.

### *Recall Task Method*

Each participant engaged in three iterations of the recall task. Each iteration began with the participant trading the study sentences list with the experimenter for a new list of sentences identical to the study set, but with one term from each sentence replaced with a blank, and the order of the sentences randomized. The participant's task was to correctly fill-in each of the blanks with the missing word. The participant was given as much time as he or she needed to do so. The experimenter then recorded the number of correct responses and for each error, provided the correct missing word to the participant. The participant was then given the opportunity to review the study set again. The three iterations were balanced such that each term from each sentence in the study set was replaced with a blank exactly once. After the third iteration the recognition phase began.

### *Recognition Task Method*

The recognition task was conducted on a computer using the Experiment Builder software package from SR Research. Sentences were presented one at a time, centered on a 17" CRT monitor (in black font on a white background). Participants judged whether each presented sentence was a member of the study set, or not. To respond, participants hit either the z-key or the /-key, respectively. Accuracy and reaction time were recorded for each trial. After each trial, the screen blanked for 1 second, and then the word "READY" appeared for 1 second to prepare the participant for the next trial.

The participant was presented with 96 test sentences, which consisted of three exposures to each of the study set sentences, and 48 foil sentences which were not from the study set. Participants were told that they should consider sentences from the study

set to be *true*, while all others should be considered *false*. Each false sentence was generated by replacing one of the three terms from a true sentence with another term from the same category (e.g., color, thing, or place) and with the same fan. For example, for a true sentence like “The blue hat is in the garage”, one false counterpart might be “The green hat is in the garage”. Each true sentence was used to generate three different false sentences. Thus, for each exposure of a true sentence there was a corresponding false test sentence with the identical fan.

## Results

The data from one participant was excluded from the analysis below. This participant’s recognition reaction time was significantly longer than all the other participants by a large margin ( $P < .001$ ). The results below reflect the data collected from the remaining 26 participants.

### *Human Recall Performance*

Performance in the recall task improved, on average, with each of three iterations. Table 14 presents the mean number of correct responses (out of 16), the standard deviation, and the accuracy measured as a percentage for each of the three iterations of the recall phase.

This result is important because an intended purpose of the recall task was to confirm that the participants had memorized the study set before entering the recognition phase. By the end of the third iteration the participants were correctly completing the sentences 91.4 percent of the time.

Table 14. Recall accuracy

	Iteration		
	1	2	3
Correct (/16)	10.9	13.4	14.6
SD	3.7	3.1	1.8
Percentage	68.1	83.8	91.4

#### *Human Recognition Performance*

Overall, participants' accuracy and reaction time results were consistent with the fan effect (see tables 15 and 16). For both true and false sentences, accuracy was negatively correlated with sentence fan. Also, accuracy was poorer for false sentences than for true sentences for all sentence fans ( $ps < .05$ ).

Table 15. Recognition accuracy (%)

Sentence fan	Accuracy	
	True	False
3	97.5	95.5
6	95.1	91.7
9	92.1	86.3
12	82.7	77.6

Reaction time increased with sentence fan ( $p < .001$ ) for both true and false sentences, and true sentences were judged more quickly than false ones ( $p = .001$ ).

*Table 16. Recognition reaction time (ms/char)*

Sentence fan	True		False	
	Reaction time	SD	Reaction time	SD
3	59.0	18.5	64.1	19.9
6	63.6	20.0	69.3	22.6
9	74.2	21.8	86.2	31.1
12	91.3	31.5	102.5	43.6

Table 16 shows the reaction times (ms/char) for both true correct (i.e., the test sentence was true and was judged correctly) and false correct sentences of each fan.

*Table 17. Pairwise comparisons for correct reaction times*

Sentence fans	P (one-tail)
3 versus 6	0.129
6 versus 9	< 0.001
9 versus 12	< 0.001

Figure 29 shows the mean reaction times, measured in ms/character, for both true correct and false correct sentences, for each sentence fan with confidence intervals.

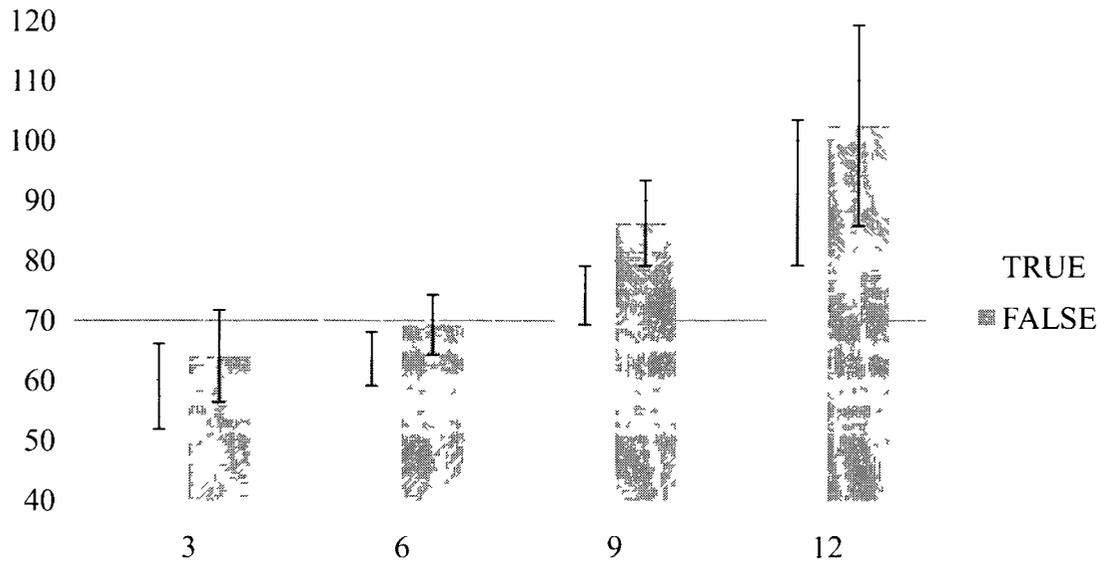


Figure 29. Recognition reaction time by sentence fan (ms/char) with confidence intervals.

There was no interaction of truth and fan ( $p = .199$ ). Table 17 presents the pairwise comparisons across fan using the Bonferonni adjustment.

#### *In Summary*

The results of the experiment confirm the fan effect as a robust phenomenon that generalizes from sentences with two content terms (Anderson, 1974) to sentences with three content terms (present research). Future work will examine whether statistically significant differences can be found in the relative contributions of the terms to the fan effect (e.g., does the *color* term contribute differently than the *thing* or *place* terms).

## DSHM

The memory modeling system used to model the described experiment was Dynamically Structured Holographic Memory (DSHM) (Rutledge-Taylor & West, 2008). DSHM is based on the BEAGLE model of the lexicon (Jones & Mewhort, 2007). The details of the DSHM architecture and the similarities between BEAGLE and DSHM can be found elsewhere (Rutledge-Taylor & West, 2007).

For an account of the use of DSHM to model the classic fan effect, and a comparison to ACT-R (Anderson & Lebiere, 1998), see Rutledge-Taylor and West (2008). For those unfamiliar with DSHM, a brief introduction follows.

DSHM makes use of holographic reduced representations (HRR) to encode knowledge in memory. See Plate (1995) for a discussion of the sort of HRRs used by DSHM (and BEAGLE). A DSHM system is composed of a collection of items that are represented internally as two vectors of numbers: i) the environmental vector is static and uniquely identifies the item in the system; ii) the memory vector is dynamic and encodes all of the associations an item develops with other items. The lengths of these vectors are fixed for an instance of DSHM, but can be initially set to any positive integer which is a power of 2.

DSHM takes collections of items as input (called complex items; collections of items are items themselves). The structure of a complex item can be expressed using left and right brackets. For example the sentence “The red hat is in the garage” can be expressed [red:hat:garage]. The system can also allow items to have a hierarchical structure. Here, the context tags used to classify an item as background knowledge (false) versus experimental knowledge (true) applies to the sentence as a whole, and so is

up a level in the hierarchy, expressed: [true [red:hat:garage]]. Items can bear ordered (delimited by colons) or unordered (delimited by spaces) relationships with one another.

Information is extracted from DSHM by presenting it with incomplete complex items. For example, a query for the color of an item might be expressed [true [?x:hat:garage]. Any missing items are called query items and in DSHM syntax are always preceded with a question mark, (e.g., “?x”). A query item is like a variable that DSHM is tasked with resolving. DSHM makes use of information stored in the memory vectors of the provided items to generate a rank ordered list of candidate items for replacing the query item. Each candidate completion is accompanied by a numerical value ranging from 0.0 to 1.0 that indicates the strength of the completion. This strength is referred to as the confidence (i.e., how confident DSHM is in the completion being correct, or appropriate). It can also be thought of a context relative activation value, to use an ACT-R term.

A DSHM model is constructed by making choices about how information is represented in complex items, what vector size should be used, what training regime is used, and what sorts of queries are presented to the system.

### The Model

Twenty-seven simulated participants were run (to correspond to the 27 human participants). It was found that a range of vector lengths allowed the simulated participants to produce reasonable recognition accuracy and reaction time results. However, fitting the recall data was more of a challenge. Uniformly using a vector length of 64 produced significantly poorer performance than the average for the human

participants, while a vector length of 128 produced significantly better results. No value in between is possible (vector length must be powers of 2). In order to produce good average scores, nine of the simulated participants were given memory systems that made use of vector lengths of 64, while the other 18 used vector lengths of 128.

### *Study Phase*

Prior to learning the study set, each simulated participant read 1026 background knowledge sentences, each encoded as a flat ordered list of three content terms associated with a tag ‘false’; “[false [color:thing:place]]”. The false sentences included either one or two of the content terms appearing in the study set. The remaining one or two terms were nonsense terms that did not occur in the study set sentences. The background knowledge was needed in order to give the simulated participants some basis for making errors. Without background knowledge there is nothing for DSHM to confuse the study set sentences with; DSHM does not make use of explicitly added noise.

The simulated participants read each sentence in the study set once or twice (to account for the differences in how well the human participants prepared themselves for the first task) prior to beginning the recall phase. Sentences from the study set were associated with a context tag representing ‘true’; “[true [color:thing:place]]”.

### *Recall Performance of the Model*

Like the human participants, the simulated participants produced responses to fill-in the blank questions in the recall phase. For example, “The \_\_\_\_\_ hat is in the garage” was submitted to the DSHM participant as “[true [?:hat:garage]]”. The system outputs a list of candidate responses, in rank order. The one with the highest rank was considered

to be the simulated participant's response. If the system's response item matched the correct missing term, the trial was scored as correct.

After each iteration the DSHM participant read each of the study set sentences once for every three incorrect responses on the previous iteration. The majority of human participants took the opportunity to review the study set, even after scoring perfectly on the previous iteration. Thus, the DSHM participants re-read the study set a minimum of once between trials.

Table 18 presents the recall accuracy for the simulated participants. Although, the accuracy plateaus after the 2<sup>nd</sup> iteration, there is an overall good match for accuracy and standard deviation, as demonstrated in figure 30 (only the standard deviations for the human data are shown).

*Table 18. Model recall accuracy*

	Iteration		
	1	2	3
Correct	11.1	14.1	14.1
SD	3.2	2.1	1.9
Percentage	69.4	88.2	88.2

#### *Recognition Performance of the Model*

The simulated participants were each tested on the same 96 test sentences as the human participants. In order to produce a truth judgment the simulated participant was

presented with a query of the form “[?x [color:thing:place]]”. If the system produced ‘true’ as its highest ranked completion candidate, the simulated participant was considered to have judged the sentence to be *true*, otherwise, the simulated participant was considered to have judged the sentence to be *false*.

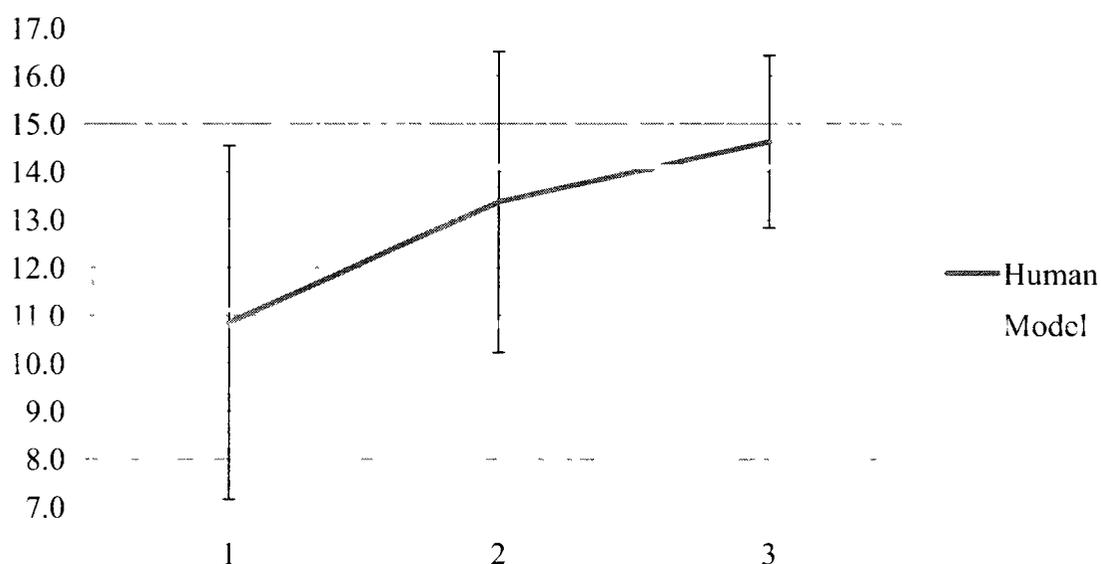


Figure 30. Recall accuracy (out of 16).

To determine the reaction time for the response, the model evaluated the degree to which the test sentence as a whole (without a context tag) (“[color:thing:place]”), resonated within the system. The sentence’s resonance is produced by a built-in DSHM method, which essentially determines how closely associated the terms in the sentence are to one another. Here, the resonance value is interpreted as indicating how familiar the sentence seems to the simulated participant. Thus, if the sentence is judged to be true, a high resonance should make this decision easier. If it is lower, it should make the

decision harder. The opposite is the case for judgments of false. It should be difficult to reject a sentence that seems familiar, and vice-versa.

The formula used for translating resonance to reaction time was  $RT = 32 / R$ , where R is the value provided by the memory system of the simulated participant, and RT is reaction time measured in ms per character. For true sentences R is the resonance value for the sentence. For false sentences, R is the resonance value for the sentence subtracted from an upper limit on resonance values. This upper limit was estimated to be the maximum resonance value calculated for any of the true sentences (0.64). Table 19 presents the reaction time data for the model.

*Table 19. Model reaction time (ms/char)*

Sentence fan	Reaction Time	
	True	False
3	61.1	67.4
6	69.0	69.5
9	79.6	77.1
12	87.8	94.5

Figure 31 presents a comparison of the human and model data for correct trials. The solid lines correspond to the human data; the dashed lines correspond to the model data; the light lines correspond to the true data; and the dark lines correspond to the false data.

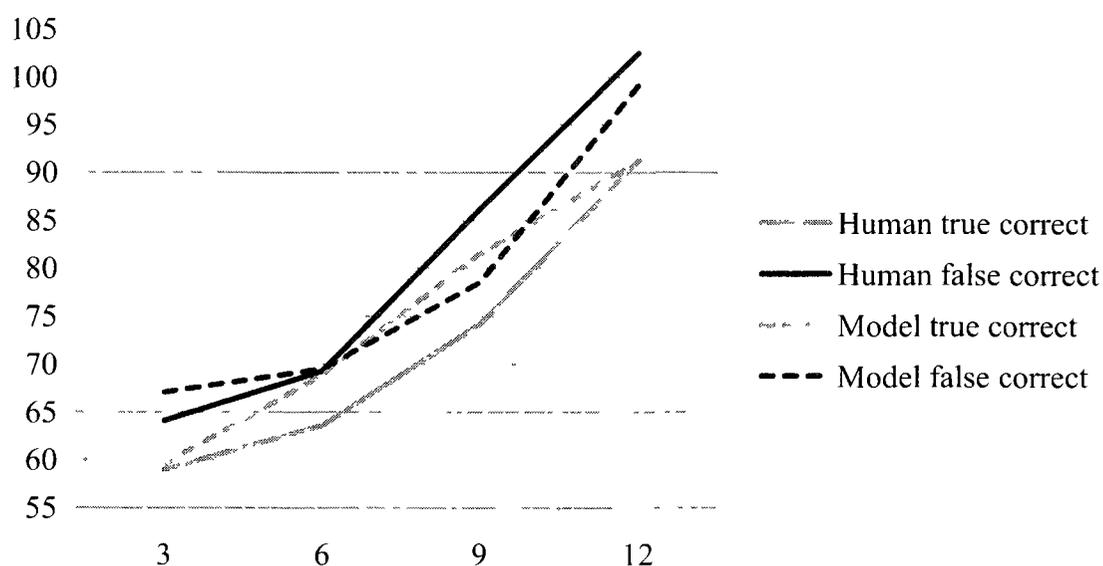


Figure 31. Recognition reaction times.

In terms of judgment accuracy, the model outperformed the human participants. The simulated participants had a judgment accuracy of 100% for true sentences and 98.5% for false sentences. It is possible that this discrepancy may be due to the relatively small body of ‘interfering’ background knowledge in the simulated participants relative to real human participants.

#### In Summary

In general the model results provide a good match to the human data, in that 1) the false sentences take longer, on average, to affirm or deny than do true sentences (77.1 ms/char versus 74.4 ms/char); 2) a fan effect is observed for both true sentences and false sentences; 3) the model provided a good fit to recall performance as well as recognition performance; and 4) the formula used to convert raw model output to reaction time values

is simple and provides a good fit to the recognition times using a single scaling parameter.

#### On-Going Work: Effect of How Fan is Distributed?

Part of the motivation for this experiment and model construction was to investigate whether each content word in a sentence contributes equally to the difficulty in recognizing a sentence as true (i.e., a member of study set). It was hypothesized that the color term may make a smaller contribution to the fan effect than the thing or the place. This is because the color terms are adjectives and more ubiquitous than the things or places, which are nouns. However, whether the thing or the place should carry more weight was not predicted given conflicting intuitions about why one or the other should be more influential. For example, the thing term might be the most influential because an object's type (e.g., hat) is a more intrinsic property than its location (or color). Alternately, place might be more influential: Grammatically, the color and thing share a common noun phrase, while the place does not share its prepositional phrase with any other content word.

The human data were not clear cut with regard to the influence how fan was distributed among content terms. By fan distribution, we are referring to the possible pattern of the fans of the words making up sentences with a particular fan. There are three different ways to make fan 6 sentences (color term fan = 1, thing = 1, place = 4; 1,4,1; 4,1,1), and three ways fan 9 sentences (1,4,4; 4,1,4; 4,4,1), while there is only one way to make fan 3 sentences (1,1,1) and one way to make fan 12 sentences (4,4,4).

No significant effects of fan distribution were found among fan 6 sentences. But, among sentences with a fan of 9, an ANOVA with revealed that fan distribution did have an impact on RT ( $p = .002$ ). Specifically, RT was faster when either the thing or place was unique (i.e., fan 1) and slower when the color was unique. Put another way, when trying to judge whether a sentence is true (e.g., “The red hat is in the garage”), knowledge of other objects with the same color (red ball) adds less difficulty than knowledge of other items of the same type (hat) or other items in the same place (garage). Further, RTs tended to be faster when the thing type was unique rather than the location, though this trend did not reach significance.

Note: a simple variation in the representation of sentences in DSHM would be able to account for this effect because DSHM is capable of representing facts that have hierarchical structure. In fact, DSHM already leverages this capability in the current model. In the representation “[true [color:thing:place]]”, the three term sentence as a whole aggregate is the hierarchical sibling of the context tag (‘true’). In order to represent sentences where the thing term is dominant, the color and place need only be embedded in a list of peripheral properties as in the following representation: “[true [thing:[color:place]]]”.

Exploratory simulations confirm that using this type of representation predicts differences in reaction times among fan 9 sentences, where the thing fan dominates the fans of the other two terms. Similarly, for sentences with an overall fan of six and a thing fan of four are significantly slower than fan 6 sentences with a color fan of four, or a place fan of four. Additional human testing is required to gather more information about the effects of fan distribution. But it is noteworthy that such hierarchical effects could be

naturally afforded by structural aspects of a DSHM architecture. This line of research is on-going.

## Appendix

### *Relationship To the Two Term Model*

Rutledge-Taylor and West (2008) presented a model of the fan effect, as described in Anderson (1974). This model provided a good match to the human data, but used a different mechanism for calculating recognition accuracy and reaction time values, than the one presented here. This mechanism, which we will refer to as the ‘retrieval’ mechanism operates as described below.

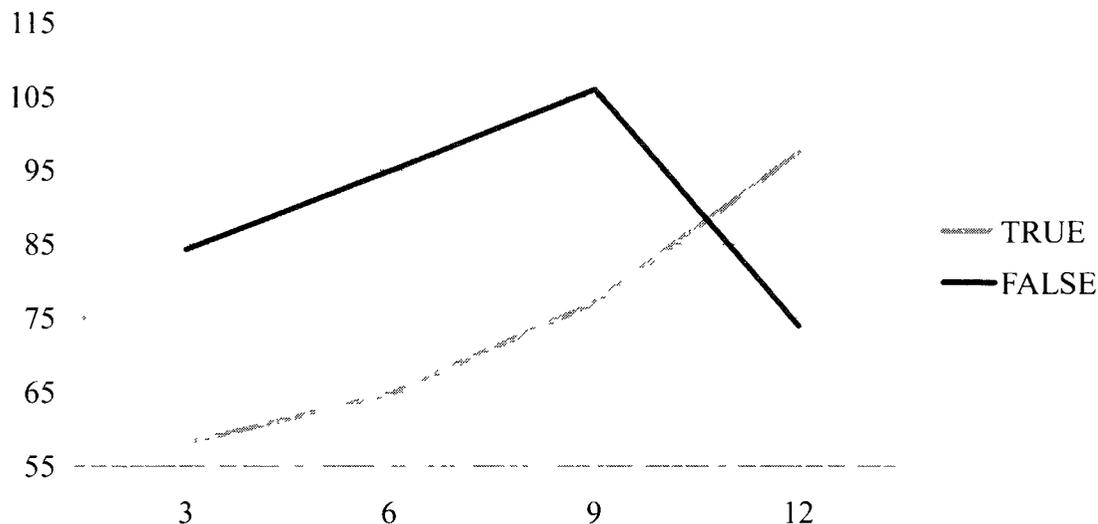
Whether DSHM recognizes a sentence, or not, according to the retrieval mechanism is based on how strongly the words in the sentence are associated with one another. Specifically, if at least one of the words in the sentence (referred to as a target word) can be recovered using the other words in the sentence as cues, the sentence as a whole is recognized (as true), otherwise, it is not.

If the sentence is recognized, the reaction time is based on strengths (e.g., confidence values) of the recovered target words, which are high on average, resulting in low reaction times. If the sentence is not recognized, the reaction time is based on strengths of the words that were retrieved (but did not match the target words). On average the strengths of these retrieved words are lower, resulting in higher reaction times. Additionally, the fans of the words in the sentences affect the strengths of the retrieved words and it these strengths that are the basis for the fan effect in the DSHM model.

*Using the Retrieval Mechanism in the Three Term Model*

The retrieval mechanism for generating recognition and accuracy results for the DSHM model was initially tested on the current stimuli and without using background knowledge sentences, which are not necessary for this mechanism. The retrieval mechanism produced a 100% accuracy rate for identifying true sentences, but only a 36% accuracy rate for rejecting false sentences.

The retrieval mechanism produced a very good fit to the human true correct reaction times, including the characteristic exponential curve observed in the human data (for both trues and falses). However, the model results for false correct (i.e., correct rejections) reaction times were drastically different from that of the human data; see figure 32.



*Figure 32.* Reaction times (ms/char) using the retrieval mechanism.

The explanation for the model's false correct data has to do with the number of true near neighbors the false sentences have. Here, 'near neighbors' are defined as two sentences that differ only by a single word. The number of near true neighbors a false sentence has is correlated with its fan. This is the result of the counter-balancing of true and false sentences. The existence of near neighbors makes little difference in the recognition results for false fan 3, 6 and 9 sentences. However, for fan 12 sentences there are true near neighbors that are retrieved (for each target word) with very high strengths. This results in low reaction times for false sentences with a fan of 12. For example, when presented with the false sentence "the black mug is in the garage", the true sentence "the grey mug is in the garage" is retrieved with a high confidence value, when internally testing to see if "[?x:mug:garage]" retrieves 'black' as a candidate completion of the query term '?x'.

Due to the failure of the retrieval mechanism to provide a satisfactory account of the reaction times for correct rejections, the new mechanism described above was developed. It is the authors' belief that the retrieval mechanism ought to work for most DSHM models under most circumstances. However, in cases such as the one presented here, the new mechanism can be applied in order to generate recognition reaction times for correct rejections that are resistant to the effects of true near neighbors.

#### References

- Anderson, J. R. (1974). Retrieval of propositional information from long-term memory. *Cognitive Psychology*, 6, 451-474.

- Anderson, J. R., & Lebiere, C. (Eds.) (1998). *The atomic components of thought*. Mahwah, NJ: Lawrence Erlbaum.
- Jones, M. N., & Mewhort, D. J. K. (2007). Representing word meaning and Order information in a composite holographic lexicon. *Psychological Review*, *114*, 1-37.
- Plate, T. A. (1995). Holographic reduced representations. *IEEE Transactions on Neural Networks*, *6*, 623-641.
- Rutledge-Taylor, M. F. & West, R. L. (2007). MALTA: Enhancing ACT-R with a holographic persistent knowledge store. In D. S. McNamara & J. G. Trafton (Eds.), *Proceedings of the 29<sup>th</sup> Annual Conference of the Cognitive Science Society* (pp. 1433-1438). Nashville, TN: Cognitive Science Society.
- Rutledge-Taylor, M. F. & West, R. L. (2008). Modeling The fan effect using dynamically structured holographic memory. In B. C. Love, K. McRae, & V. M. Sloutsky (Eds.), *Proceedings of the 30<sup>th</sup> Annual Conference of the Cognitive Science Society* (pp. 385-390). Washington, DC: Cognitive Science Society.

A HOLOGRAPHIC MODEL OF FREQUENCY AND INTERFERENCE:  
RETHINKING THE PROBLEM SIZE EFFECT

Previously published as Rutledge-Taylor, M. F., Pyke, A. A., & West, R. L. (2010). A holographic model of frequency and interference: Rethinking the problem size effect. Carleton University Cognitive Science Technical Report 2010-02. URL <http://www.carleton.ca/ics/TechReports>

Contribution

This paper helps demonstrate that the fan effect is robust and can occur in areas of cognition outside of the classic paradigm of linguistic fact memorization. In this case, it is shown that interference effects contributing to the problem size effect can be explained as a special case of the fan effect.

This paper also demonstrates that DSHM can model frequency effects. That is, when a fact is studied more frequently than another fact, it is recalled more quickly than a fact that has been reinforced less frequently. Frequency in ACT-R is accounted for in the base-level activation of chunks.

Abstract

In this paper we used a holographic memory system to model Zbrodoff's (1995) findings on the problem size effect, a well-known effect in the area of math cognition. The data showed the effects of manipulating both frequency and interference. We successfully modeled this using DHSM (Rutledge-Taylor & West, 2007), which has previously been used to model the fan effect (Anderson, 1974; Rutledge-Taylor & West,

2008). This demonstrates that frequency and interference effects arise naturally as a function of how holographic systems work.

### Introduction

The Dynamically Structured Holographic Memory system (DSHM) uses holographic representations as a way of modeling human memory. It is based on Jones and Mewhort's BEAGLE lexicon model (Jones & Mewhort, 2007). The details of DSHM and the similarities to BEAGLE are discussed in Rutledge-Taylor and West (2007). One function that DSHM models well is memory interference. Rutledge-Taylor and West (2008) showed that the *fan effect* (Anderson, 1974) falls naturally out of the DSHM architecture.

The *fan effect* is a term used to describe a memory phenomenon in which the time needed to verify a fact is related to the number of other facts in memory that include concepts in common with the target fact (Anderson, 1974). The fan refers to how many facts share memory elements with the target. For example, if a person's declarative memory contained three propositions: "the hippie is in the park", "the lawyer is in the store", and "the lawyer is in the bank", then the fan of the terms 'hippie', 'park', 'store', and 'bank' are one, while the fan of the term 'lawyer' is two. As first demonstrated by Anderson (1974), larger fans cause slower reaction times in human participants. This result is consistent with the theory that similar facts cause interference in the retrieval process.

The DSHM model of the fan effect (Rutledge-Taylor & West, 2008) is conceptually similar to the ACT-R model of the fan effect (Anderson & Reder, 1999).

Specifically, the emergent behaviors of the DSHM model can be interpreted as being consistent with the spreading activation mechanisms used to produce the fan effect in ACT-R. In brief, DSHM makes use of Holographic Reduced Representations (HRRs) to encode associations between concepts. According to the DSHM model, memory is composed of holographic items. Each item consists, primarily, of two large vectors: an *environmental vector* and a *memory vector*. The environmental vector is static after its creation. It is used as the system's representation of the identity of the item in memory. In contrast, the memory vector is dynamic. The memory vector of an item is used to store all the associations between the item and other items in the system.

Associations between items are formed when a set of items is given to the system as input. Many details aside (see Rutledge-Taylor & West, 2008), the memory vector of each item in the input becomes more similar to the environmental vectors of the items it is associated with. Since the vectors can be thought of as coordinates in a high dimensional space, this means that the memory vectors of items move closer to the environmental vectors of items they are associated with. The fan effect results from the encoding process. Items with larger fans get *pulled* in more different directions and this impedes them from moving particularly close to specific vectors. In the example above, the lawyer gets pulled toward both the store and to the bank, whereas the other characters move closer to a single location. In DSHM the inverse of the distance from the question vector to the nearest vector is interpreted as the activation level of that vector and, similar to ACT-R, the activation level determines the speed of retrieval.

The fan effect, proper, addresses only the effect of inter-fact 'interference' on the efficiency of fact retrieval. However, there is another factor that also strongly impacts

retrieval speed/efficiency: the person's frequency of exposure to that fact. For example, if a participant reads "the lawyer is in the store" once and "the lawyer is in the bank" four times, the fans of 'store' and 'bank' are each still one. However, one would expect that the association between 'lawyer' and 'bank' to be stronger than the association between 'lawyer' and 'store'. Thus, both fan effects and frequency effects impact the efficiency of fact retrieval.

In ACT-R, frequency of exposure is represented separately by the base level activation function (Anderson & Lebiere, 1998). In DSHM frequency produces an effect by causing a fact to be pulled more in one direction than another. For example, if the lawyer was in the store more often than in the bank, the vector representing the lawyer would be end up closer to the store vector than the bank vector. To test the interaction of frequency and fan in DSHM we modeled the data of Zbrodoff (1995), who manipulated both of these in the context of studying arithmetic cognition (i.e., retrieving simple facts such as " $2 + 3 = 5$ ").

### Zbrodoff's Experiments

In arithmetic cognition research, it is often found that small sums, like  $2 + 3 = 5$ , are more quickly retrieved than large sums, like  $5 + 7 = 12$ . This is the so-called *problem-size effect* (reviewed by Zbrodoff & Logan, 2005). It is known that small problems are presented more frequently in math texts than large problems (Hamann & Ashcraft, 1986), so this effect could be due to frequency of exposure, however, interference between memory elements has also been proposed as an explanation (Seigler, 1987; Vergats & Fias, 2005). Zbrodoff's (1995) goal was to investigate the

extent to which different retrieval times for different math facts should be attributed to fan effects (which she termed 'interference') or frequency effects (which she termed 'strength') or an interaction of these two effects.

Zbrodoff (1995) conducted four experiments that manipulated the effects of strength and interference to assess their relative contribution to the problem-size effect. In each experiment participants were shown mathematical problems with a potential answer on a computer screen. The participant's task was to press one key if the problem was correct, and press a different key if the problem was false. To manipulate frequency and interference for facts in memory, and to eliminate pre-experimental practice effects, instead of using regular arithmetic, Zbrodoff's stimuli were alphabet arithmetic facts (e.g.,  $A + 3 = D$ , which indicates that the number three letters past A is D). The first addend was always a letter of the alphabet; the operator was always addition; the second addend was 2, 3 or 4; and, the sum was a letter of the alphabet. The problem was considered true if translating the letters to numbers according to their index in the alphabet resulted in a true math fact. For example, " $A + 2 = C$ " is true because translating 'A' to 1 and 'C' to 3, results in the true math fact " $1 + 2 = 3$ ". Participants were told that they could determine whether a problem was true or false by starting with the first addend (e.g., A) and then counting through the alphabet the number of letters specified by the second addend (e.g., 3).

In each experiment, participants were exposed to large blocks of problems, each of which consisted in repeated instances of a group of 12 unique problems. Each group consisted of six true and six false problems. A False problem was a problem for which the answer was incorrect.

In Experiments 1 and 2, each problem consisted of the combination of one of six letter addends with one of the digit addends (2, 3 or 4). Each letter was paired with only one digit, and each digit was paired with two letters. False problems were generated by setting the incorrect answer to one letter past the correct answer (e.g.,  $B + 3 = F$ ). The groups of problems were counterbalanced so that each letter addend was paired with each numerical addend equally frequently Table 20 provides an example of a group of problems for Experiments 1 and 2.

Groups of problems for Experiments 3 and 4 were similar to those for Experiments 1 and 2 with two changes. In each stimulus set only two unique letter addends were used (e.g., A and B). Each letter was paired with each of the three digit addends. The false answers were either one letter past the correct answer or one letter before it. Table 21 provides an example of a group of problems for Experiments 3 and 4.

*Table 20.* Experiment 1 & 2 group of problems

Letter addend		Number addend		True answer	False answer
A	+	2	=	C	D
B	+	3	=	E	F
C	+	4	=	G	H
D	+	2	=	F	G
E	+	3	=	H	I
F	+	4	=	J	K

### Re-Analysis of Zbrodoff's Experiments

Zbrodoff (1995) used a method of analyzing the data that was not useful for our purposes. Specifically, she plotted a straight line through the reaction times associated with the different addends and used the slope of this as an index of the magnitude of the Problem Size effect. This is a common way of analyzing the Problem Size effect within the area of Math Cognition. However, since we were interested in testing our model, not the Problem Size effect, we re-analyzed the reaction time data (which is provided in the paper) and came up with a somewhat different interpretation of the results.

*Table 21.* Experiment 3 & 4 group of problems

Letter addend		Number addend		True answer	False answer
A	+	2	=	C	D
A	+	3	=	D	E
A	+	4	=	E	F
B	+	2	=	D	C
B	+	3	=	E	D
B	+	4	=	F	E

#### *Experiment 1*

Experiment 1 was focused on learning in the short term. In it participants were presented with three blocks of problems. Each consisted of 96 true and 96 false problems. This was problematic for us because the learning process involved initially doing the calculations

to get the answers. Therefore, these blocks represent a mixture of calculating and memorizing. By the third block we assume participants were using memory, but may have still been relying on calculation as well.

Since DSHM models memorizing only, we could not represent the effect of calculating. Hence, our goal was not to model the learning curve. Instead we were focused on the long term learning trends. Because of this, we did not include Experiment 1 in our analysis. However, the results of Experiment 1 showed that frequency of exposure to a problem eventually resulted in faster reaction times, regardless of whether the addend was large or small. This is consistent with the DSHM model since, if all other things are held equal (as they were in Experiment 1), more exposures results in faster recall.

### *Experiment 2*

The purpose of Experiment 2 was to determine whether the frequency effect demonstrated in Experiment 1 held up once performance had reached asymptote. That is, with enough practice does response time performance converge despite differences in frequency of exposure. To test this, participants were presented with 15 blocks of problems (3

blocks per day) identical to those from the *Standard* condition of Experiment 1, which was designed to mimic real world math learning conditions where smaller numbers are encountered more frequently than larger numbers. To accomplish this real world problem frequency, the addend 2 problems were presented 24 times per block, the addend 3 problems were presented 16 times per block, and the addend 4 problems were presented 8 times per block.

The results showed that performance did converge as it reached an asymptote. However, we also noticed that reaction times were much lower at asymptote in Experiment 2 than in Experiments 3 and 4 (approximately 600 msec in Experiment 2; 1000 msec in Experiments 3 and 4). To explain this discrepancy we examined the stimuli and found that in Experiment 2 each letter-answer was uniquely associated with a different letter-addend in the question. For example, the addend 'A' is associated only with the sum 'C' (see Table 20). Therefore participants could memorize a pairing between the letter-addend in the question and the letter-answer. Given that this was not the case in Experiments 3 and 4 we feel the learning and use of this strategy is a likely explanation for the faster reaction times in Experiment 2.

#### *Experiment 4*

We begin our modeling results with Experiment 4, because it is less complex than Experiment 3. In Experiment 4 all of the problems in Table 21 were presented with equal frequency. To model this, each problem, including the answer and whether the answer was true or false, was encoded as an item and presented to a DSHM model (see figure 33).

[true [A: 2:C]]

*Figure 33.* An example DSHM item representing an addition fact.

Each item was unordered and was composed of two sub-items: an atomic item representing the truth of the problem, and a complex item representing the problem. The

item representing the problem was ordered and consisted of atomic items representing the first addend, second addend, and the sum. Each item was presented to the DSHM model once for every exposure by the human participants to the corresponding problem.

There were two ways the model could decide if a question was true. Once was to give the model a complete representation of the problem, and have it decide whether it was true or not. To do this, the model was presented with an incomplete item with a structure similar to that in figure 33, but with the truth item replaced with a query item; see figure 34.

[?truth [A:2:C]]

*Figure 34.* An example DSHM item used for recalling the truth of an addition fact.

The model would then return a completion with the query item replaced with either true or false. The second way was to submit the question with the answer replaced with a query item and the truth component set to true; see figure 34.

[true [A:2:?sum]]

*Figure 35.* An example DSHM item used for recalling the correct sum of an addition fact.

In this case the model would return what it believed to be the correct answer (note, the model can make errors but this data is not presented here). The second method

fit the data better than the first, suggesting that when presented with a problem, the participants were determining what the correct sum should be and making a comparison of this value to the sum presented. In this case the model makes the same predictions for true and false questions. Consistent with this, the human reaction times for the true and false questions were very similar. To get accurate reaction times from the model the inverse of the confidence values for the completions (i.e., activation levels) were scaled up by a factor of 400. Note that this represents a claim that the confidence values produced by the model translate directly into reaction times.

Figures 36 and 37 present the human and model results, respectively. Note that the reaction time for the addend 3 questions is slowest in both the human data and the simulation. Zbrodoff (1995) concluded that there were no differences between the three addend conditions. This was because she assumed that only a linear trend in the relationship between addend and reaction time was possible, and none was found. However, the higher reaction times for the addend 3 problems was due to a real quantifiable effect, and was predictable. The reason for non-linearity was because of a fan effect type pattern in the stimuli. The problems with a numerical addend of 3 had a greater fan (7) than did the problems with addends of 2 or 4 (fan of 6). Because the relationship between addend and reaction time is not linear taking the slope of the reactions times, by addend was inappropriate and occluded the results.

We can also see that model learns faster than the human participants and very quickly asymptotes. As noted above, this is because the model does not calculate the answers. To avoid making the graphs too small only the first six blocks are presented. However, after 6 blocks the human data was at asymptote.

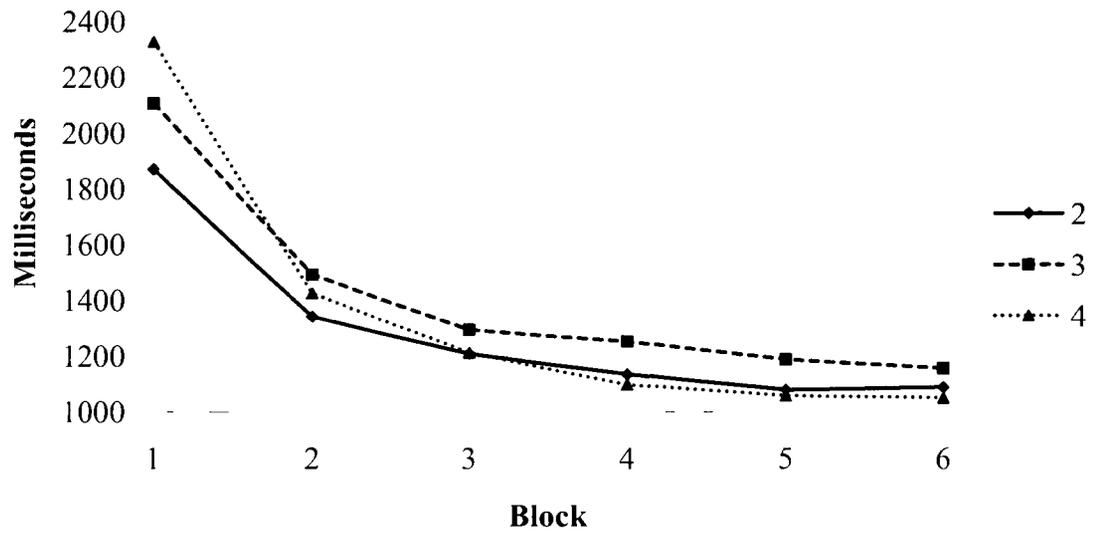


Figure 36. True reaction times for human participants by addend in experiment 4.

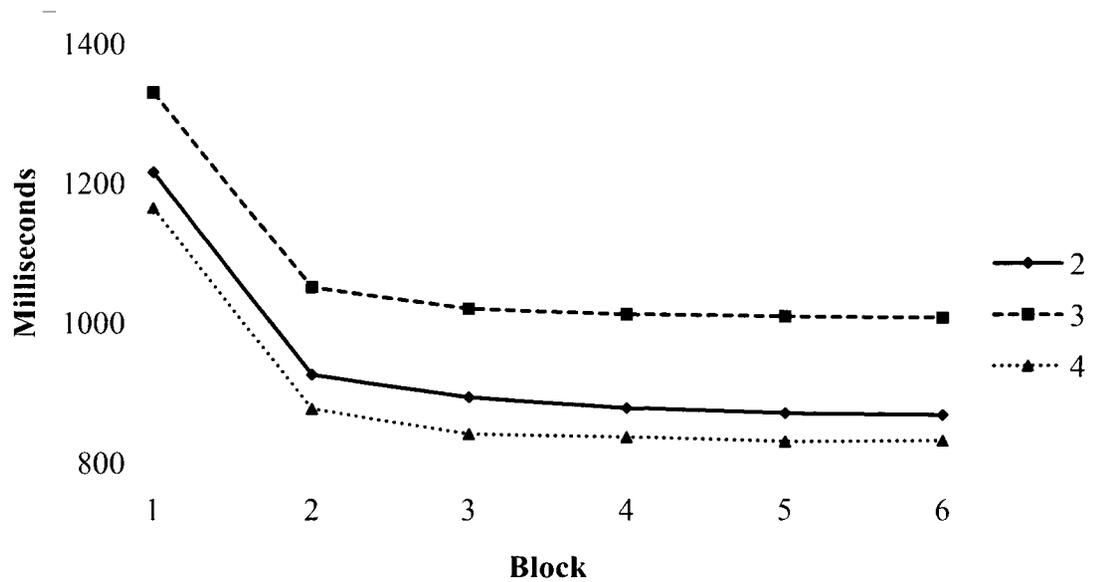


Figure 37. True reaction times for the model in experiment 4.

### *Experiment 3*

Experiment 3 was the same as Experiment 4 except that frequency was manipulated in the same way as in Experiment 2. That is, the questions with the smaller numerical addends were presented more frequently. The model used here was exactly the same as the one used to model Experiment 4, where only the fan was manipulated; no parameters were altered.

Figures 38 and 39 shows the human data and the simulation results. Overall, the model does a good job of accounting for the results. The most impressive aspect of this is the fact that the model predicts that the addend 4 problems ought to have the slowest recall in early blocks, due to the frequency effect, but that with time, the fan effect, described in the discussion of experiment 4 (above), takes over in later blocks causing the addend 3 problems to become the slowest. An exception to the good fit of the model occurs in the later blocks (not shown on the graphs) where the model continues to exhibit a fan effect, while in the human data the addend 4 reaction times converge with the addend 3 reactions times. This difference is difficult to interpret. It could be that the model does not predict well for long term learning, although it did accurately predict long-term learning for Experiment 4. Another possibility is that participants were using a rehearsal strategy between sessions. If participants were recalling the questions and checking them by calculation, or rehearsing them, it could produce this effect since the addend 4 questions would be harder to recall due to the low frequency of presentation (for random recall without a cue, interference should not play a role). Therefore, the addend 4 questions would not be practiced as much. Given that the model fits well in every other respect, further tests using other sources of human data need to be done.

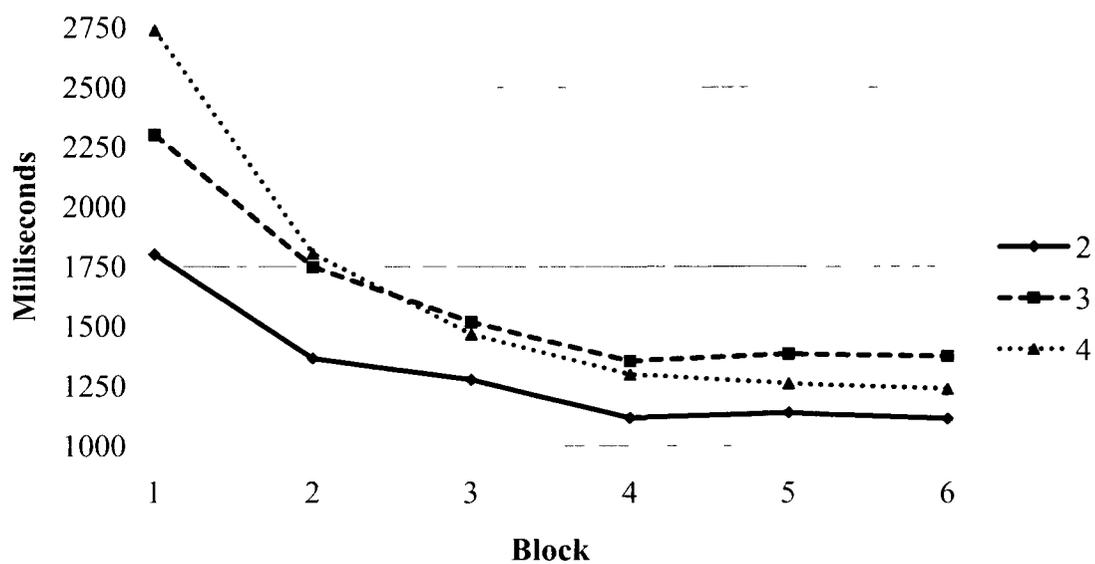


Figure 38. True reaction times by addend for human participants in experiment 3.

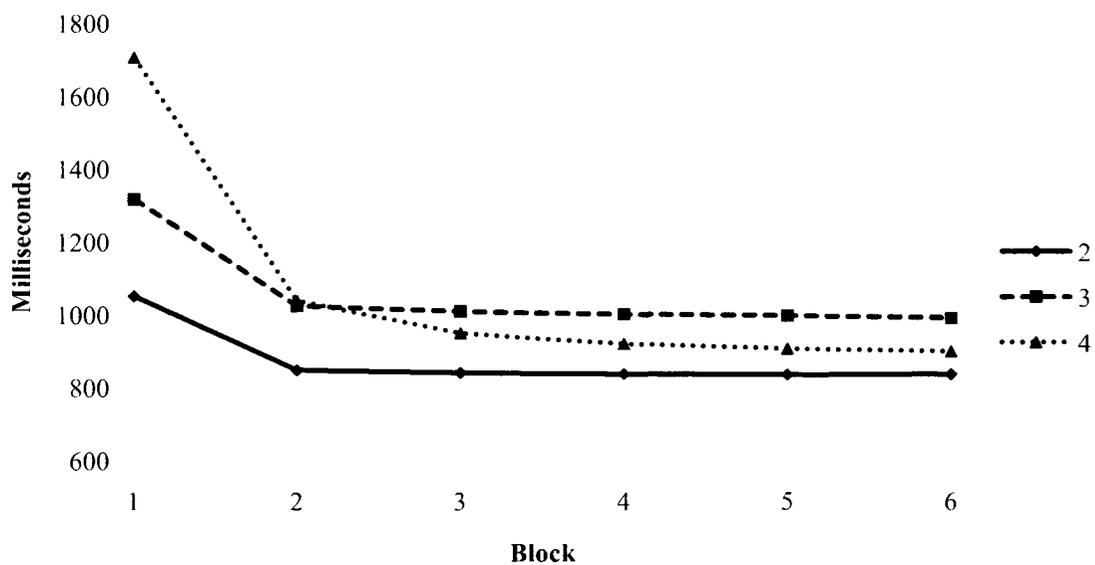


Figure 39. True reaction times by addend for the model in experiment 3.

## Conclusions

Frequency and interference are key predictors of human memory performance (e.g., retrieval time) in many cognitive tasks. The individual and joint effects of these factors are clearly evidenced in arithmetic fact learning. The present simulations demonstrated that the DSHM modeling system is not only well suited to capture interference (fan) effects (see also Rutledge-Taylor & West, 2008) but also frequency effects in conjunction with fan effects. Both of these effects arise naturally out of the holographic system, suggesting that this way of representing memory is in some way analogous to the way memories are represented in the human brain. Our results also suggest that cognitive modeling may be an appropriate tool in the area of math cognition.

## References

- Anderson, J. R. (1974). Retrieval of propositional information from long-term memory. *Cognitive Psychology*, 6, 451-474.
- Anderson, J. R., & Lebiere, C. (Eds.) (1998). *The atomic components of thought*. Mahwah, NJ: Lawrence Erlbaum.
- Anderson, J. R. & Reder, L. R. (1999). The fan effect: New results and new theories. *Journal of Experimental Psychology: General*, 128(2), 186-197.
- Hamann, M. & Ashcraft, M. (1986). Textbook presentations of the basic addition facts. *Cognition and Instruction*, 3, 173-192.
- Jones, M. N., & Mewhort, D. J. K. (2007). Representing word meaning and Order information in a composite holographic lexicon. *Psychological Review*, 114, 1-37.

- Rutledge-Taylor, M. F. & West, R. L. (2007). MALTA: Enhancing ACT-R with a holographic persistent knowledge store. In D. S. McNamara & J. G. Trafton (Eds.), *Proceedings of the 29<sup>th</sup> Annual Conference of the Cognitive Science Society* (pp. 1433-1438). Nashville, TN: Cognitive Science Society.
- Rutledge-Taylor, M. F. & West, R. L. (2008). Modeling The fan effect using dynamically structured holographic memory. In B. C. Love, K. McRae, & V. M. Sloutsky (Eds.), *Proceedings of the 30<sup>th</sup> Annual Conference of the Cognitive Science Society* (pp. 385-390). Washington, DC: Cognitive Science Society.
- Siegler, R. (1987). The perils of averaging data over strategies: An example from children's addition. *Journal of Experimental Psychology: General*, *116*, 250-264.
- Verguts, T. & Fias, W. (2005). Interacting neighbours: A connectionist model of retrieval in single-digit multiplication. *Memory & Cognition*, *22*, 1-16.
- Zbrodoff, N. J. (1995). Why is  $9 + 7$  harder than  $2 + 3$ ? Strength and interference as explanations of the problem-size effect. *Memory and Cognition*, *23(6)*, 689-700.
- Zbrodoff, N. J. & Logan, G. D. (2005). What everyone finds: The problem-size effect. In J. I. D. Campbell (Ed.), *Handbook of mathematical cognition* (pp. 331-346). New York: Psychology Press.

# DYNAMICALLY STRUCTURED HOLOGRAPHIC MEMORY FOR RECOMMENDATION

Previously published as Rutledge-Taylor, M. F., Vellino, A. & West, R. L. (2010). Dynamically structured holographic memory for recommendation. Carleton University Cognitive Science Technical Report 2010-01. URL <http://www.carleton.ca/ics/TechReports>

## Contribution

This paper supports the claim that DSHM models can operate on large knowledge bases (tens of thousands of items). This ability is essential for building models that contain an amount of general knowledge equivalent to what a normal adult human has acquired in his or her lifetime. An important contribution that this paper makes is in describing ‘the cluster method’, which is the only example of memory vector comparison in this portfolio.

## Abstract

Dynamically Structured Holographic Memory (DSHM) is a cognitive model of associative memory that can be applied to the problem of recommendation. DSHM uses holographically reduced representations to encode the associations between objects that it learns about to generate recommendations. We compare the recommendations from this holographic recommender to a user-based collaborative filtering algorithm on several dataset, including MovieLens, and two bibliographic datasets from a scientific digital library. Off-line experiments show that the DSHM recommender predicts movie ratings

as well as collaborative filtering and much better than collaborative filtering on very sparse bibliographic data sets. DSHM also has a unified underlying model that makes multi-dimensional recommendations and their explanations easier to develop. However, DSHM requires significant amounts of computational resources to generate recommendations and it may require a distributed implementation for it to be practical as a recommender for large data sets.

### Introduction

The function of a recommender system is to recommend items (such as songs, books, movies or merchandise) that are likely to be of interest to a user given both the preferences of the user and the collective preferences of the user community. Recommender systems have been used not only to enhance personalized e-commerce web sites (Pierrakos, Paliouras, Papatheodorou & Spyropoulos, 2003), but also to offer a richer information retrieval experience in digital library portals (Frias-Martinez, Magoulas, Chen & Macredie, 2006).

Most conventional recommender systems operate by clustering similar items according to some characteristic of the item (content-based recommendation; Pazzani & Billsus, 2007), by measuring the similarity among ratings that users have given to items (collaborative filtering – either memory-based or model-based; Adomavicius & Tuzhilin, 2005) or by combining the two in some manner (hybrid recommenders; Burke, 2002). Hybrid recommenders have been used as a strategy in situations where pure collaborative filtering suffers from well-understood limitations, for instance in situations where usage or rating data is sparse.

Data sparsity is especially problematic in the context of recommending journal articles in a digital library, where a relatively small number of scholars (users) need recommendations from among a relatively large number of articles (items). Extremely small user-item ratios demand more than collaborative filtering alone can provide (Vellino & Zeber, 2007)

In addition, recommender systems need to provide explanations to the user about how the recommendation was made. This allows the user to ascertain the relevance of recommendations, assume greater control over how the recommender behaves and have greater confidence in the validity of its results. However, recommenders that use multiple sources of information and integrate results from multiple algorithms generate relatively ad-hoc explanations. Hybrid recommendation algorithms are typically more difficult to generate explanations for than ones that have a unified prediction model (Burke, 2002).

This paper describes an approach to recommendation based on a cognitive model of associative memory – Dynamically Structured Holographic Memory (DSHM). This approach is motivated by the intuition that applying a cognitive model of memory could enhance the effectiveness of an information retrieval system by making it behave more like a human expert. In Huggett, Hoos and Rensink (2007), Michael Hugget et. al. assert that “to make information management systems more useful to a wider range of people, it seems reasonable to apply functional cognitive principles to data storage and retrieval”.

Thus, the motivation for the experiments described below is grounded in the question of whether a recommender system can be made to perform more like a human expert. Often the best way to get a movie recommendation is to ask the video store clerk

to recommend a movie based on what you have enjoyed viewing recently. Similarly, your best bet for finding relevant journal articles is to ask an expert in the field what to read next given a set of articles that you have found useful. Our objective was not so much to discover a recommendation technique that was more effective or efficient for typical recommender tasks in commercial applications as to verify the intuition that a cognitive model of memory was a viable alternative to purely statistical or probabilistic approaches.

We believe that a holographic cognitive model of memory has two noteworthy features that make it suitable for addressing the recommendation problem. The first is adaptability. A holographic memory model can be given any item of information - even properly encoded visual cues - and can potentially make use of it. The second, which follows from the first, is novelty. Human beings, with their wide variety of knowledge sources are often able to integrate information in a way that produces a novel result. For a recommender system, this could mean generating serendipitous, but potentially useful and otherwise unlikely recommendations in ways that could extend beyond the serendipity provided by collaborative filtering systems.

Measuring the accuracy of a recommender on bibliographic data is difficult because there are no standard offline benchmarks for testing recommender quality. Furthermore, for recommending scholarly articles, end-user satisfaction is a better overall measure of a recommender's success than are the relevance or precision of its underlying algorithms (Torres, et al.). Hence we chose first to benchmark DSHM with off-line

experiments on the MovieLens data, which is often used to benchmark CF recommenders.

We then performed the off-line experiments described below to compare CF and DSHM on two very sparse bibliographic datasets taken from a digital library. These experiments do not directly evaluate the serendipity of the recommendations provided by DSHM - this is a characteristic that we intend to study in future work. Here we are solely concerned with demonstrating that, as a starting point, a DSHM based recommender can perform with competitive accuracy to existing CF systems.

### *Collaborative Filtering*

Recommender systems typically operate on three kinds of entities: users, items and the preference ratings that users have assigned to items. Given a set of ratings for certain items – whether they are obtained from users explicitly or implicitly from, for example, browsing patterns – a user-based collaborative filtering (CF) system will attempt to predict the rating of a previously unrated item for the active user based on how other (similar) users previously rated the same item. In contrast with collaborative filtering methods, which use algorithmic statistics to generate recommendations, DSHM encodes the co-occurrence of a set of items in the representation of the items themselves and uses the memory model of the items' history of associations to generate recommendations.

As noted in the introduction, recommending journal articles in a digital library is more problematic than recommending other kinds of items because the usage data is sparse relative to the number of items in the collection (Huang, Chen & Zeng, 2004). One remedy for this problem is to use bibliographic citations as a proxy for user ratings

(Torres, McNee, Abel, Konstan & Riedl, 2004). This was the technique we used for the experiments described in this paper.

In the experiments with CF we used a user-based CF recommender that implements k-nearest neighbour and cosine correlation in the Taste framework (now part of the ApacheLucene machine learning library Mahout, see <http://lucene.apache.org/mahout>). Note that, in this instance, where user preferences are equivalent to article citations, a user-based approach is equivalent to an item based one.

### *DSHM - Dynamically Structured Holographic Memory*

DSHM is a cognitive model of human long-term memory (Rutledge-Taylor & West, 2007; 2008). It is designed as a tool for understanding how the human mind organises knowledge, e.g., how it stores, confuses, forgets and accurately retrieves information. The implementation of DSHM used for the experiments described in this paper is a self-contained Python program that does not depend on any other modeling software. DSHM has also been reimplemented in Java. The Java version includes improvements in caching vector computations and in system state persistence.

DSHM makes use of Holographic Reduced Representations (HRRs) to encode associations between concepts (Plate, 1995). It is based on Jones and Mewhort's BEAGLE model of the lexicon (Jones & Mewhort, 2007). DSHM generalizes BEAGLE to apply to any memory type. BEAGLE can be considered a special case of DSHM. According to the DSHM model, memory is composed of holographic items, which we will refer to here as H-items. H-items represent entities to be remembered by the DSHM

system. We refer them as “H-items” to avoid confusion with the generic term “item” used to refer to rated entities in a recommender.

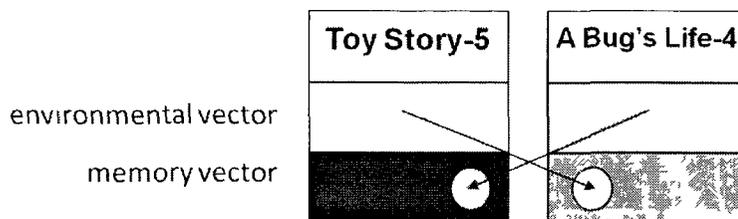
Each H-item consists, primarily, of two large vectors - an “environmental vector” and a “memory vector” - of floating point numbers, each with a Euclidean length of 1.0. The numbers that make up the vectors are generated at random, and adhering to a Gaussian distribution. The environmental vector is static (i.e., it does not change) after its creation. It is used as the system’s internal representation of the mental entity corresponding to the H-item. In contrast, the memory vector is dynamic (i.e., it changes over time). The memory vector of one H-item is used to store all the associations between that H-item and other H-items in the system. The number of elements in the vectors determines the memory capacity of the system. This is because the greater numbers of elements results in less significant collisions between unrelated vectors, on average. Simple models of memory phenomena often use vectors with 128 or 256 elements. The experiments described in this paper used very large data sets. Thus, the number of elements in each vector was set to 2048, providing the models with adequate memory capacity to perform the recommendation task. The drawback to using large vectors is that they take more computational resources to manipulate mathematically. The relationship between vector size,  $n$ , and the computational cost is  $n \ln(n)$ .

Associations between H-items are formed when a set of H-items is given to the system as input. From a cognitive perspective this can be interpreted as the H-items co-occurring in a thought, a verbal utterance or a perception. The system distinguishes between sets for which the order of the elements is essential to the content of the set as a

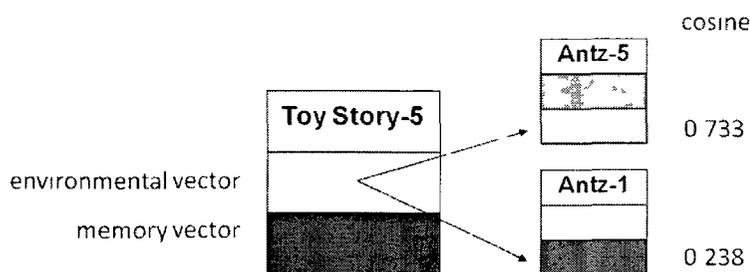
whole (e.g., the words in a sentence) from those that are not (e.g., the things scattered about my work desk that I am perceiving right now). If the set of H-items is unordered, every H-item is associated with every subset of the other H-items in the set, up to a predefined maximum number of elements. In the experiments presented in this paper, this maximum was set to the lowest permissible value (namely one); i.e., each element of a set is only associated with every other element of the set, but not with any combinations of pairs and other n-tuples of elements. The effect of increasing this maximum is to improve the context sensitivity of the system at the cost of additional computational resources. A typical DSHM model of memory (Rutledge-Taylor & West, 2008), applied to smaller data-sets, would use a value of two or three.

These associations between elements are recorded by binding the environmental vectors of the H-items in each subset together, and adding the resulting vector to the memory vectors of the other H-items in the set (see Figure 40). A binding is formed by recursively computing the circular convolution of the environmental vector of an H-item from the given subset and the binding of the remainder of the subset. The circular convolution of vectors is commutative, and thus the order in which the H-items of an unordered set are bound together does not affect the resulting aggregate binding. However, if the set is an ordered list, the neighbours of every H-item up to a system-defined maximum distance are associated with the given H-item in a manner that preserves the order of the H-items (Rutledge-Taylor & West, 2007). The result of these methods of associating H-items together is that the memory vector of each H-item encodes information about all of the other H-items with which it has co-occurred. The

strength of the association between two items can be determined by calculating the cosine similarity between the memory vector of one item and the environmental vector of the other (see Figure 41).



*Figure 40.* Learning associations in DSHM.



*Figure 41.* Item Similarity in DSHM.

Conceptually, DSHM shares properties in common with, but distinct from computationally inspired cognitive architectures such as ACT-R and neural networks that make use of unsupervised learning. We have argued elsewhere that DSHM may help bridge incompatibilities between these two general frameworks (Rutledge-Taylor & West, 2008).

*The structure of DSHM*

A collection of H-items in a DSHM system defines a multi-dimensional state-space where each environmental and memory vector is a point on the surface of a hypersphere with radius 1.0. This state-space implements a complex semantic network occupied by the items represented in the system. One property of this organisation of H-items is that given an incomplete pattern, the provided, known, H-items from the pattern can be used to predict the most likely candidates for completing the pattern. This is done first by generating a set of “probe” vectors based on the memory vectors of the known H-items in the pattern. Each probe is computed by reversing the binding process described above and predicts an environmental vector that approximates the vector of the item that best completes the pattern (Rutledge-Taylor & West, 2007). We refer to this method of prediction as “decoding”.

These probes are compared to the environmental vectors of the H-items that might be completions of the pattern. Each candidate H-item in the system is then ranked according to a score based on the sum of the cosines of the candidate H-items’ environmental vector and each of the probes. The H-items with the greatest combined scores are those proposed by the system as the most likely completions of the pattern.

By leveraging this pattern-completion property, DSHM has been successfully applied to modeling human memory recall and recognition (Rutledge-Taylor & West, 2008).

Another interesting property of the organisation of H-items in the system is the relationships between H-items’ memory vectors. When a large set of patterns has been

entered into the DSHM system, and the associations between H-items have been computed, the memory vectors of the H-items in the system will cluster. These clusters are usually open to a meaningful interpretation relating to the content represented by the H-items in the sets of patterns originally presented to the system. For example, as is the case with BEAGLE – where the patterns are sentences and the items are words – the H-items will cluster according to semantic similarity, or synonymy (Jones & Mewhort, 2007). The reason for this is that items, which are in some way equivalent and can be interchanged for one another, will tend to have the same sets of neighbours, and will therefore develop similar memory vectors. Thus, given an H-item, its memory vector can be used as a probe to be compared to other H-items' memory vectors. The matches found will be the H-items that are similar to the one providing the probe. We refer to this method of prediction as “clustering”. It is important to emphasize that the similarity between two H-items discussed here is not based on any content about the items provided to the DSHM system. Rather, the system is inducing the similarity of H-items based only on the patterns in which the items occur. This ability is, of course, part of what make DSHM an interesting cognitive model of memory.

The following section describes how DSHM can exhibit collaborative filtering effects by making use of these properties of DSHM systems.

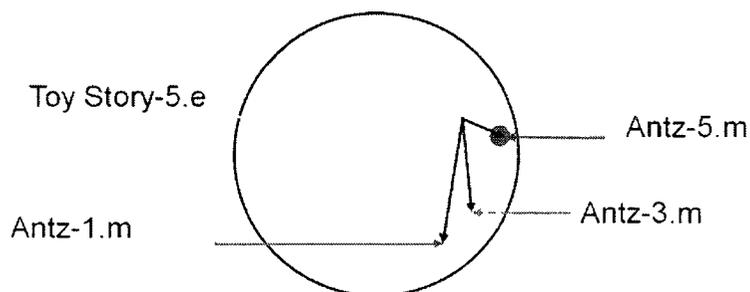
#### *Recommendation in DSHM*

Given a set of items (e.g., books, movies, or journal articles), and a set of users who are defined by what subset of the items they have rated, the purpose of a CF system is to accurately predict what rating a user is likely to assign an item that he or

she has not yet rated. Given a test item and a test user, CF assigns to the item a value based on two factors. The first (in user-based CF) is the similarity of the test user to a neighbourhood of other users who have also rated the item and the second is the ratings assigned to the item by the other users. There are various functions that can be used to compute this value. What such a function provides is a metric for how consistent the test item is with the items rated by the test user. In the absence of any other considerations, the concept of “user” is inessential and serves only as a container for his or her rated items.

In contrast, CF methods which use algorithmic statistics to generate recommendations, the most natural way to recommend items in DSHM, in a manner equivalent to user-based CF, is to treat a user as simply a set of unordered ratings. The ratings are considered to be unordered because we assume that the value given to an item is highly, although not completely, independent of any considerations that may impose an order on the ratings, such as the dates the ratings were provided. We define a rating as the combination of an item and a preference value. When imported into the DSHM system, the ratings are converted into H-items, which we will refer to as H-ratings, and are associated with one another by the binding process described above. Thus, for each H-rating, information about other H-ratings with which it has co-occurred is stored in the memory vector of the H-rating. Once all of the users’ ratings have been imported into the system, the state-space defined by the H-ratings’ memory vectors will cluster as described above.

Hence, H-ratings that occur in similar sets of ratings will be located together. This does not only mean that ratings by the same user will be located together in the state-space (this may or may not be the case). It is also the case that two ratings that have never been rated by a common user can have very similar memory vectors if the users who have rated them have other ratings in common. Thus, given a rating, the memory vector of the corresponding H-rating can be used to locate other H-ratings that are consistent with the given rating (see Figure 42).

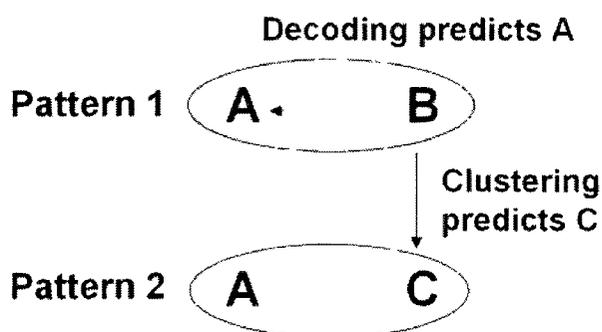


*Figure 42.* DSHM state space.

Therefore, to predict a new rating, the combined influence of all of the known ratings by the test user can be used to converge on a set of probes that point to a location in the state space where DSHM calculates the new rating ought to exist. H-ratings occupying this location are then returned by the system as its predictions.

It should be noted that, strictly speaking, in DSHM, only the decoding method corresponds to traditional user-based CF. A simple example illustrates the difference between the decoding and clustering methods (see Figure 43). Consider an object A

that co-occurs with another object B in one instance, and with a third object C in another instance. The decoding method, when applied to B, will predict A. This is because only A co-occurs with B. The clustering method, when applied to B will predict C because only C has the same neighbours (i.e., A) as B. We have included reference to the clustering method in this paper because it makes use of the exact same data as does user-based CF and produces noteworthy results



*Figure 43.* Decoding versus clustering in DSHM.

## Experiments

Which methodology to choose for evaluating a recommender's performance depends a great deal on the kind of user task for which the recommender is applied, the datasets being used to perform the evaluation as well as what characteristics (e.g., accuracy, usefulness, serendipity) of the recommender are being evaluated (Herlocker, Konstan, Terveen & Riedl, 2004). For this study, our objective was to compare DSHM with a conventional CF recommender to understand both whether DSHM is an applicable technique and how it compares in accuracy with CF on

sparse bibliographic datasets. We first establish a baseline for DSHM on the MovieLens dataset and then extend the comparison between these two approaches to two different datasets obtained from a repository of biomedical journal articles.

### *MovieLens*

Our first experiment was to establish a baseline of predictive accuracy comparisons between DSHM and CF on the MovieLens dataset. This test was done using the usual 10%-90% cross-validation methodology. Previous studies of CF on this data show a Mean Absolute Error (MAE) of approximately 0.73, depending on parameters for the neighbourhood size (Sarwar, Karypis, Konstan & Riedl, 2001).

The greatest challenge to predicting ratings using DSHM, is the fact that the current implementation has no innate ability to represent magnitude. Given that the goal is to predict a numerical rating, some means of accommodating magnitudes needed to be incorporated into the representation of ratings in the system. We decided that for every movie in the MovieLens dataset, five distinct H-items would be created, one for each possible rating, e.g., “Toy Story (1995)” with a rating value of 4 would be a single atomic entity in the system, as would “Toy Story (1995)” with a rating value of 5.

The cognitive interpretation of this is that the mental representations that correspond to liking a given movie very much, and liking it only somewhat, differ in many dimensions, and not just on a single numerical scale. Hence, representing each rating for each movie as entirely different H-items presumes nothing about how the ratings for the same movie ought to be related to one another. Thus, prior to the

learning phase, the various H-ratings representing the ratings of a movie did not bear and a priori relationship to one another. That is, the H-rating representing a movie X, with a preference value of 4 was no more related to the H-rating representing X with a value of 5 than the H-rating representing X with a value of 1. Given this method of representing ratings, a rating prediction then becomes the task of determining which of the five H-ratings for a given movie is most highly associated with the H-ratings corresponding to a test user's other ratings.

Of the 6040 MovieLens users, approximately 10% (614) were randomly removed from the sample and used as test users. The DSHM recommender was trained on the ratings provided by the remaining 5426 users. For each of the test users, the DSHM recommender made predictions for ten of the test user's ratings. These predictions were done one at a time, and used all of the user's other ratings as sources from which to base the predictions.

### *Results*

As mentioned above, there are two distinct, but related, ways in which DSHM can predict a rating. In the case of the decoding method, DSHM is being asked to find ratings that are likely to have co-occurred with the known ratings. Here, DSHM produced a MAE of 1.23. This poor result was initially surprising. The decoding method had been employed with a great deal of success in a DSHM model of human memory data (Rutledge-Taylor & West, 2008). We hypothesised that the poor performance of the model was due to our choice of how to represent ratings. The drawback of not presuming any relationship between the pair of ratings

corresponding to a given movie with values of 4 and 5, which CF can leverage, is obvious. By creating five H-ratings for each movie, there are too many items relative to the number of users for DSHM to discover reliable co-occurrence patterns of preferences for the test users.

In contrast of the performance using the decoding method, the clustering method produced a competitive MAE of 0.71. In this latter case, DSHM is being asked to find ratings that are similar to the known ratings. The reader is reminded that the similarity discussed here is based only on what is induced by the DSHM system, and not based on any content explicitly provided about the movies. This task is more resilient to noise because the value of an H-rating's memory vector is the accumulated influence of many associations, which, on average, represents a reliable location in the state-space occupied by relevantly similar ratings. Note that this method of measuring H-rating similarity would allow for clusters to represent movie preference profiles. For example, a romance movie rated 1 could cluster with an action movie rated 5.

The authors are confident that if the provided with a sufficiently large set of training data the decoding method ought to produce better results. This is because the decoding method is potentially more powerful than the clustering method, but requires either less noisy data or data with more complete coverage of the data space. The potential of the decoding method pertains to its ability to identify more serendipitous patterns in data.

### *Journal Articles*

Our second experiment is modeled after the off-line experiments in TechLens+, which evaluated the effectiveness of different strategies for recommending journal articles in a digital library context (Torres et al., 2004). As with TechLens+, we treated articles as “users” and articles’ lists of references as lists of boolean “ratings” for other articles (although we note that while bibliographic references in an article are an indicator of relevance they are not necessarily an indication of *favourable* relevance in the mind of the author).

Our experiment compares DSHM and CF on Top-N results on two bibliographic datasets: one was extracted from a collection of 31,000 articles from 39 Medicine journals and the other from a collection of 114,000 articles from 107 Biology journals. The Medicine collection was reduced to 7495 articles by eliminating articles for which references were unavailable. In addition, the references we used to populate the preferences matrix were only the references that were made to articles in that collection. Overall, the total number of references in the collection was over 273,000, but only 4100 of them were references to articles in the collection, for an average of only 0.55 references per article. In other words, the collection was both very sparse and very loosely connected. The Biology collection was reduced to 38,667 and also had a small average number of references (1.15 per article) to articles in the collection. The connectivity of the article collections — measured as the number of references in an article plus the number of articles that cite it — was slightly above 1 for the Medicine collection

and slightly above 2.3 for the Biology collection, as compared to 14 in the CiteSeer collection used in TechLens+ (Torres et al., 2004).

Our experimental method also differs from the TechLens+ study in some respects. One is that the cross-validation was not 10-fold and not random. Instead, we chose to perform leave-one-out evaluations exhaustively on a sample of the articles biased towards those with the most references to items in the bibliographic collections. One reason for using this strategy rather than the random selection strategy was that the likelihood of picking a random article with only one or fewer references to articles in the collection was quite high. Leaving one reference out for each of the articles with the most references seemed more likely to produce a recommendation that was correct. Thus we chose a subset of 95 test articles in each collection which had between 17 and 5 references per article in the Medicine collection, for a total of 570 prediction attempts and between 32 and 13 references in the Biology collection, for a total of 1491 prediction attempts.

For this experiment, the DSHM implementation was essentially the same as for the MovieLens experiment, except that there needed to be only one H-item representation for each unique article. For each of the non-test articles, the H-items representing the article's references were associated together. To make a prediction, the memory vectors of the H-items corresponding to the remaining references were used to generate probes used to rank all of the other articles in the collection according to how highly they were associated with the probes. Again, the two distinct methods of generating these probes, as described above, were used. The

DSHM recommender was asked either to recommend articles that were most likely to have co-occurred with the provided references, or to find references that were similar to the provided references.

### *Results*

The results of these experiments are summarized in Figures 44 and 45. In the case of the journal article recommendations, the DSHM recommender produced very good results via the decode method, presented here. Unlike for the MovieLens data, the clustering method did not produce better results for journal article recommendation.

In the case of the Medicine data set, DSHM correctly predicted the first item (Top-1) 76 times out of 570; made a Top-5 recommendation 145 times; Top-10 161 times; and, top-30 178 times. In the case of the Biology data set, the totals were: 127 Top-1; 296 Top-5; 371 Top-10; 443 Top-30, out of 1491 total recommendations. In contrast, our traditional CF system's performance was somewhat poorer. In the medicine case the totals were: 28 Top-1; 90 Top-5; 108 Top-10; and 131 Top-30. In the biology case the totals were 51 Top-1; 144 Top-5; 210 Top-10 and 340 Top-30. Comparing these results, it appears that DSHM does considerably better than CF at making Top-1 and Top-5 recommendations, but that the performance of the two systems converges at the tail end. DSHM also seems to be more sensitive to the reference structure of the two article collections and performs proportionately better with the Medicine collection.

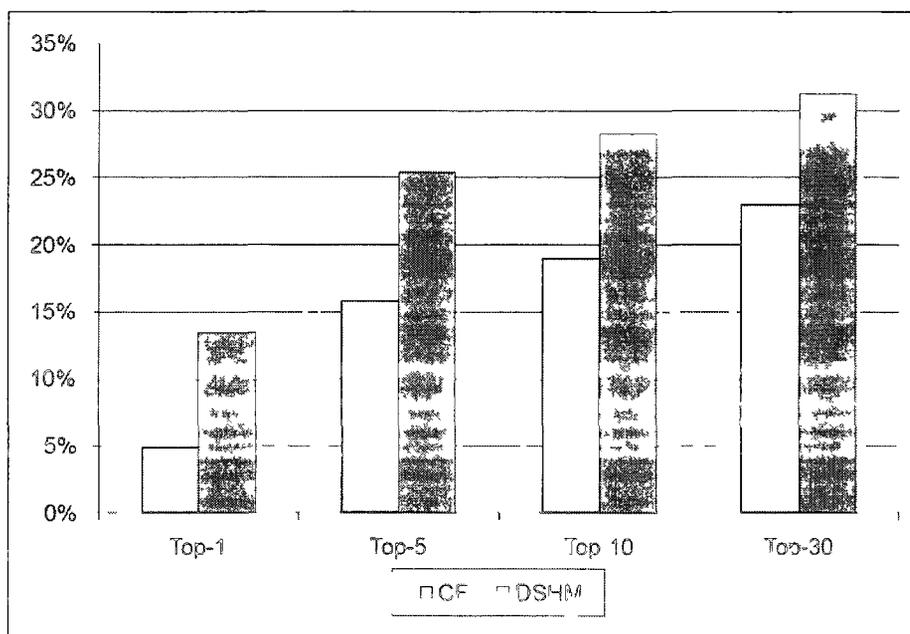


Figure 14 Top N results for CF and DSHM on medicine collection

In other words, DSHM correctly predicted the Top-1 references in the Medicine collection almost 3 times as often as CF (76 versus 28), and converged to a 36% improvement over CF for the Top-30. For the Biology collection the accuracy of DSHM is a little less dramatic but exhibits the same trends, e.g., 127 Top-1 predictions for DSHM versus 51 for CF, and a 30% improvement for Top-30.

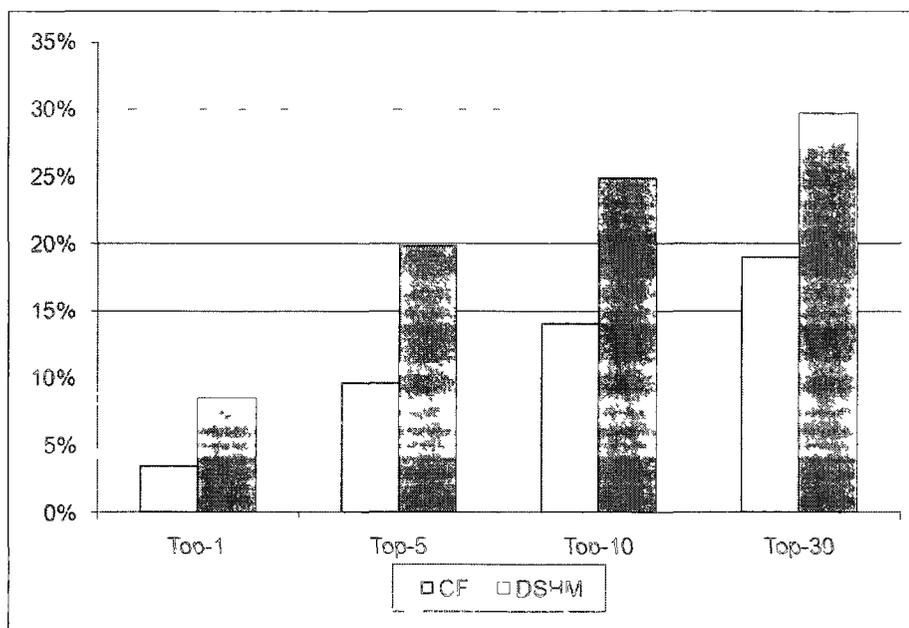


Figure 45 Top-N results for CF and DSHM on biology collection.

### Conclusion

Predicting bibliographic citations from a holographically reduced representation of bibliographic information shows that recommending Top-N items offers better accuracy than CF on very sparse datasets. We interpret this superiority of DSHM as resulting from its ability to self-organise based on the information extracted from the data.

In addition to being more accurate for sparse datasets, the flexibility of holographic recommenders offers promising possibilities for recommending items that have ratings in multiple dimensions as well as item correlations that are content-based. DSHM provides a unified mechanism for implementing what would otherwise be considered a “hybrid” recommender.

The benefits of using DSHM as a recommender for large datasets are mitigated by the considerable computational cost of producing them. In applications where recommendations must be provided quickly, DSHM may not be able to respond fast enough. In some respects, DSHM extracts too much information from the available data, but at too great a computational cost.

In cases where this information is largely composed of noise, the computational resources required to process the information produces too small a return to justify the expense, especially in on-line applications. This is due to the cost in space and time of performing thousands of matrix computations to produce each recommendation. Hence, variants on collaborative filtering techniques are still more practical for digital library recommender systems under most foreseeable circumstances. Nevertheless, in situations where significant pre-compiling of information is feasible, e.g., for relatively static or very small datasets, or when few unique queries are made to the system (see 3), DSHM may be useful for maximally digesting the available data in advance.

#### Future Work

Our future work includes both a research and an applications development effort.

#### *Research*

Our future research on DSHM as a recommender system will focus on examining exactly how information is exploited differently in DSHM compared to

CF. This will include cluster analysis of the vector state-space of DSHM recommenders, as well as a detailed examination of how learning in DSHM differs from model building CF. We would also like to compare the accuracy of a DSHM system to which content information (e.g., movie genres or article abstracts) has been added, against the accuracy of typical hybrids of CF and content-based filtering.

In addition, we intend to compare the serendipity characteristics of DSHM recommendations. We believe DSHM may distinguish itself by mimicking the serendipity of recommendations that human experts provide and differ significantly from the serendipity of collaborative filtering.

Finally we plan to investigate the explanatory capabilities of a DSHM system and compare those explanations with ones that are derived from a hybrid CF and Content-Based Filtering algorithm. We believe that DSHM's unified representation of H-item associations provides an opportunity to generate recommendation explanations that improve upon ad-hoc hybrid explanations.

All these experiments will be developed on an open-source, Java implementation of DSHM and their performance evaluated on a Hadoop cluster (<http://hadoop.apache.org/>).

### *Applications*

DSHM is currently being used in the development of an application that provides enhanced metadata to information management systems (IMS). One of the features of this IMS is a dynamic, faceted classification system that uses metadata fields which are customized for each deployment. Facets are distinct, mutually

exclusive, and collectively exhaustive perspectives used to describe information. Faceted classification systems are commonly used in the field of Library and Information Sciences as comprehensive set of categories that can be arranged in multiple ways rather than only in a rigid hierarchy (Taylor, 2006).

One objective of this application is to reduce the information management burden placed on the end-user. Thus, unlike standard faceted classification systems, the logical relationships between the facets are built into the system. These relationships constrain the possible combinations of metadata values, thus making it easy to autocomplete the metadata fields that follow as a logical consequence. For example, once the end-user has selected his or her name in the Name field and a project name in the Project field, the system will determine that other fields, such as Function and Country, have only one possible value and these fields will be filled in automatically.

The DSHM recommender enables the application to also predict the likely, though not logically necessary, metadata values. The backend server maintains a DSHM model which takes collections of metadata field and value pairs as input. Whenever a user saves a document, the metadata assigned to it is sent to the server and the model is updated. When the user prompts the system for suggested metadata values the DSHM model is presented with a query that contains the metadata fields and values that the user has provided so far, and the remaining empty fields are queried for possible values. The DSHM model then produces rank ordered lists of possible metadata values for each of the empty fields. Each possible metadata value

is associated with a real number that represents how strongly it satisfies the empty field's relationship to the completed fields in the DSHM model.

The current application-centric research focuses on determining which metadata values for empty fields are appropriate enough to recommend to the user. DSHM will provide candidates for filling each empty field, even if they are not strong candidates. The current approach is to provide a single metadata value as a recommendation for each empty field only if the internal numerical value associated with the metadata value exceeds a certain threshold. The current best solution is to evaluate whether the value associated with the highest rated metadata value is significantly greater than the value associated with the second highest rated metadata value. If this is the case, there is sufficient certainty that the highest rated metadata value distinguishes itself from the other possible values, and that the second highest rated metadata value would not make a recommendation that is as good as or better than the highest rated metadata value.

Once the system has determined which metadata values for the empty fields make appropriate recommendations, the values are returned and the previously empty fields are filled with recommended values. Recommendations are distinguished from user-selected values by annotating the recommended values with a user interface marker such as a highlight.

The general problem of the computational cost of DSHM, mentioned above is not a problem provided the server has adequate computational resources, and concurrent recommendation requests are infrequent. This is because individual

metadata recommendations in this system are less complex than the digital library and MovieLens recommendations presented in this paper, and thus take much less time to compute. Additionally, recommendation is not a necessary feature of metadata assignment. Thus, the number of recommendation requests per user per day is low. Thus, the likelihood that the server will be able to respond to all recommendation requests in a timely manner is good for most anticipated deployments of the system.

### References

- Adomavicius, G., & Tuzhilin, A. (2005). Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *Knowledge and Data Engineering, IEEE Transactions on*, 17(6), 734–749.
- Sarwar, B., Karypis, G., Konstan, J., & Riedl, J. (2001). Item-based collaborative filtering recommendation algorithms. *WWW '01: Proceedings of the 10th International Conference on World Wide Web*, 285–295, New York: ACM Press.
- Burke, R. (2002). Hybrid recommender systems: survey and experiments. *User Modeling and User-Adapted Interaction*, 12(4), 331–370.
- Frias-Martinez, E., Magoulas, G., Chen, S., & Macredie, R. (2006). Automated user modeling for personalized digital libraries. *International Journal of Information Management*, 26(3), 234–248.
- Huang, Z., Chen, H., & Zeng, D. (2004). Applying associative retrieval techniques to alleviate the sparsity problem in collaborative filtering. *ACM Transactions on Information Systems*, 22(1), 116–142.

- Huggett, M., Hoos, H., & Rensink, R. (2007). Cognitive principles for information management: The principles of mnemonic associative knowledge (p-mak). *Minds and Machines, 17*(4), 445–485.
- Herlocker, J. L., Konstan, J. A., Terveen, L. G., & Riedl, J. T. (2004). Evaluating collaborative filtering recommender systems. *ACM Transactions on Information Systems, 22*(1), 5–53.
- Jones, M. N., & Mewhort, D. J. K. (2007). Representing word meaning and Order information in a composite holographic lexicon. *Psychological Review, 114*, 1-37.
- Pazzani, M., & Billsus, D. (2007). Content-based recommendation systems. In P. Brusilovsky, A. Kobsa & W. Nejdl (Eds.), *The adaptive web: Methods and strategies of web personalization* (pp. 325–341). Berlin-Heidelberg, Germany: Springer-Verlag.
- Pierrakos, D., Paliouras, G., Papatheodorou, C., & Spyropoulos, C. D. (2003). Web usage mining as a tool for Personalization: A survey. *User Modeling and User-Adapted Interaction, 13*(4), 311–372.
- Plate, T. A. (1995). Holographic reduced representations. *IEEE Transactions on Neural Networks, 6*, 623-641.
- Rutledge-Taylor, M. F. & West, R. L. (2007). MALTA: Enhancing ACT-R with a holographic persistent knowledge store. In D. S. McNamara & J. G. Trafton (Eds.), *Proceedings of the 29<sup>th</sup> Annual Conference of the Cognitive Science Society* (pp. 1433-1438). Nashville, TN: Cognitive Science Society.

- Rutledge-Taylor, M. F. & West, R. L. (2008). Modeling The fan effect using dynamically structured holographic memory. In B. C. Love, K. McRae, & V. M. Sloutsky (Eds.), *Proceedings of the 30<sup>th</sup> Annual Conference of the Cognitive Science Society* (pp. 385-390). Washington, DC: Cognitive Science Society.
- Taylor, A. G., (2006). Introduction to cataloging and classification. (10th ed.), Westport, CT: Libraries Unlimited.
- Torres, R., McNee, S., Abel, M., Konstan, J., & Riedl, J. (2004). Enhancing digital libraries with TechLens+. *Proceedings of the 2004 Joint ACM/IEEE Conference on Digital Libraries*, 228–236.
- Vellino, A. & Zeber, D. (2007). A hybrid, multi-dimensional recommender for journal articles in a scientific digital library. *Proceedings of the 2007 IEEE/WIC/ACM International Conferences on Web Intelligence and Intelligent Agent Technology - Workshops*, 111–114.

## USING DSHM TO MODEL PAPER, ROCK, SCISSORS

### Contribution

There are two interesting conclusions that can be drawn from this paper. The first is that it demonstrates that DSHM is able to model a dynamic short-term memory task. DSHM was designed to model long-term memory, the contents of which would be relatively stable. In contrast, success at PRS requires that relative priority be given to more recent facts about opponents' play, and that inconsistent facts can be reconciled appropriately.

The other interesting conclusion drawn from this paper is that it provides an example of a task that both DSHM and a neural network are able to model. This helps support the claim that DSHM connects to neurally inspired systems.

### Abstract

Dynamically Structured Holographic Memory (DSHM) is architecture for modeling memory. It was originally designed to account for long-term memory alone. However, the current model of Paper, Rock, Scissors (PRS) play (and Rock=2 PRS play) provides evidence that DSHM can be used to model tasks where the truth of facts change quickly and dynamically.

### Dynamically Structured Holographic Memory

Dynamically Structured Holographic Memory (DSHM) is an architecture for modeling memory. It was designed to account for how stable information is stored and

retrieved from long-term memory. DSHM does not presume that the relationships between concepts are entirely static. Rather, it explicitly accounts for how concepts can evolve to form associations of different strengths, over time. DSHM has been used successfully to model the fan effect (Rutledge-Taylor & West, 2008; Rutledge-Taylor, Pyke, West & Lang, 2010). It has also been used as the basis for a recommender system (Rutledge-Taylor, Vellino & West, 2008). In both of these applications, DSHM commits a set of static facts to memory.

In contrast, DSHM was not designed as a store for information whose relevance is short-lived and potentially contradictory to new information. This sort of information is not uncommon in strategic decision making tasks, such as those in that take place in competitive games described by game theory (VonNeumann & Morgenstern, 1944). However, the fact that DSHM was not designed to model these sorts of memory tasks does not preclude the possibility that it might be used to build successful models of human performance in these sorts of games. This possibility was examined by building DSHM models of human performance in the game Paper, Rock, Scissors (PRS) and a modified version of PRS. These models are presented herein.

### Existing Models of PRS

In order to provide some base-level of expectation for what might constitute good performance in models of PRS play, some existing models are briefly reviewed.

#### *Perceptron Models*

It has been shown that simple perceptron-like neural networks can be used to model human behaviour in standard PRS games (West, 1998; West & Lebiere, 2001).

The networks take sets of past opponent moves as input, and provide choices of next move as output. The networks are constructed such that they each have an output layer of three nodes, one corresponding to each of the three play options: paper, rock, and scissors. Each has one or more groups of input nodes. Each group includes three input nodes, one for each play option. Each input node is connected to each output node (see figure 46). The connections between nodes are assigned integer values (or, weights), which start at 0.

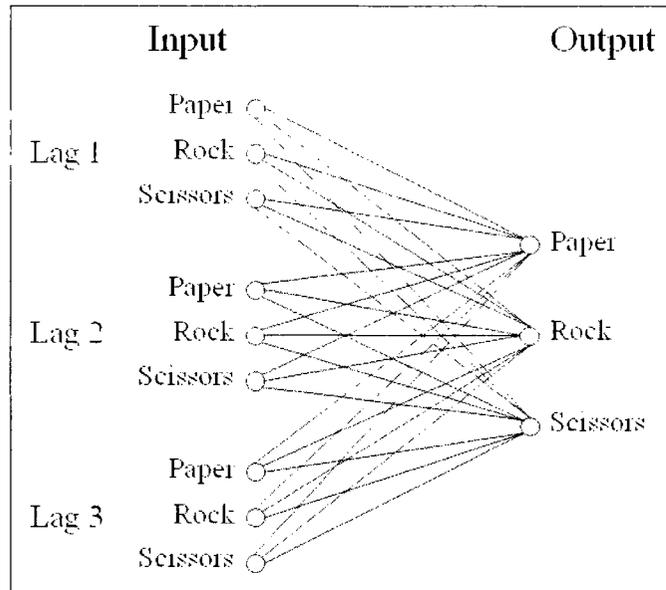


Figure 46. Perceptron networks.

If a network has only a single input group, it takes only its opponent's last move as input. To determine what option to play, the connections between the node in the input group corresponding to the opponent's move and each of the output nodes are compared. The output node attached to the connection with the greatest value determines

which move the network selects (ties are decided randomly). If the network's decision results in a win, the relevant connection is rewarded by increasing its value by one. If the result is a loss, the connection is punished by reducing its value by one. Ties are treated differently by two variations of this basic network design (West & Lebiere, 2001). Networks called 'passive' treat ties as neutral events and neither reward or punish the connection values after a tie. Networks called 'aggressive' punished connections leading to ties by 1. Only aggressive networks were tested in West (1998).

Networks with two or more input groups take a set of the opponent's last moves as input. Each additional input group beyond the first corresponds to a move further back in the opponent's play history. For example, with two input groups, one corresponds to the opponent's last move as input, while the other takes the opponent's second to last move as input. For these networks the output is determined by summing the connections between one node from each input group (corresponding to the move played on that past occasion) and each output node. Rewards and punishments are applied to all connections that contribute to the output decision. In addition to being labeled as either passive or aggressive, these networks were also labeled as 'lag 1', 'lag 2' or 'lag 3' depending on whether they attended to opponents' last one, two, or three past moves.

Both West (1998) and West & Lebiere (2001) concluded that the aggressive lag 2 network provided the best model of human PRS play. Both the humans and the aggressive lag 2 networks were able to beat the passive lag 2 (West & Lebiere, 2001), and aggressive lag 1 networks (West, 1998; West & Lebiere, 2001), by statistically significant margins. On average, humans lost to the aggressive lag 2 networks by a small

margin (West & Lebiere, 2001). However, the authors suggest that this may be due to imperfect attention and motivation on the parts of the human participants.

*Perceptron Rock=2*

The standard PRS game played by humans and network models in West (1998) and West & Lebiere (2001) were perfectly symmetrical. The three play options were identical in that each beat one of the other two moves and lost to the other; a rock versus scissors win was no different from a scissors versus paper win. In Rutledge-Taylor & West (2004), a modified version of PRS, where rock versus scissors wins were worth two points, while the other two outcomes were worth only one point each, was investigated. Ten human participants played one game against each of three network opponents. The network opponents were: the aggressive lag 1, the aggressive lag 2, and a 'rock=2' lag 1. The rock=2 network rewarded networks connections by two when it won with rock. Table 22 presents the mean points differences in the final scores of games between the human participants and each of the network models. All games consisted of 300 trials each.

*Table 22. Humans versus networks*

Network Model	Mean Pts. Diff.
Agg. Lag 1	16.5
Agg. Lag 2	5.7
Rock=2 lag 1	25.6

Two conclusions were drawn from this experiment: 1) humans were able to take advantage of the fact that wins using rock were worth two, while all three network opponents were not; 2) the rock=2 network performed the worst of all the network models due to the fact that rewarding rock wins by two became a liability (and not the anticipated advantage). This larger reward unbalanced the reward system in such a way that caused the rock=2 network played rock too frequently, and this was exploited by the human players.

An additional observation was that in Rock=2 PRS the frequencies with which player played each of the possible moves did not predict the game's final scores. For example, if two players each play paper, rock and scissors exactly 1/3 of the time each, they will tie, on average, if they are playing randomly. However, it have been demonstrated that human players (and the network models) do not play randomly (West, 1998; Rutledge-Taylor & West, 2004).

Rather than playing randomly, superior players exploit weaker opponents by predicting their opponents moves and making winning moves. As a result, they are able to achieve higher win rates than would be predicted by play probabilities alone. This effect is particularly important in the Rock=2 game, as being able to orchestrate rock versus scissors plays and to be able to avoid the opposite play is crucial to success in this game. Thus, a measure called the strategy index was invented.

The strategy index is calculated according to formula 1, below. Given two players, player 1's strategy index is player 1's total points scored minus player 2's total points, divided by the number of games played. The predicted points difference is calculated using game theory (i.e., each player is assumed to have played randomly

according to probabilities determined by the actual ratios with which the options were chosen). A positive strategy index indicates a superior ability to correctly anticipate opponent's plays and achieve a higher than probabilistically predicted number of points. The raw strategy index is relative to the number of trials per game. So, it can be also represented as a percentage according to formula 2.

$$(1) \text{ Strategy index} = \text{average actual points difference per game} - \text{predicted points difference per game}$$

$$(2) \text{ Strategy index percentage} = \text{Strategy index} / \text{number of trials per game}$$

For example, if two players play a game of 300 trials, and each player plays paper 100 times, rock 100 times, and scissors 100 times, game theory predicts that they will tie (on average). However, it is possible for one player to win all 300 trials by always matching the opponent's move with the move that beats it. In this case, the winning player would score a perfect strategy index of 300 (or 100%). In real games, a strategy index percentage of 3% or more is considered very good.

In Rutledge-Taylor & West (2004) a network model of human Rock=2 play was created by using a genetic algorithm to find a reward matrix that resulted in human like play. The criterion was to match as closely as possible the human mean points difference and the mean strategy indices against the three opponents in table 22. The resulting reward matrix was the following: rock wins = 3, paper wins = 2 scissors wins = 0; rock tie = -1, paper tie = -1, scissors tie = 0; and -3 for all losses. The performance of this model is presented below.

*ACT-R Models*

ACT-R models of both standard PRS (Lebiere & West, 1999) and of Rock=2 PRS (Rutledge-Taylor & West, 2005) have been created. Both employ an exemplar based approach and manipulate noise to establish a best fit.

For both models a chunk type with four slots was used. The isa slot was tagged with PRS to indicate that it was a PRS relevant chunk. The three remaining chunks encoded a sequence of three moves, by the model's opponent: lag0, is the opponent's current, or predicted move; lag1 is its previous move; and, lag2 is its second to last move. An example is illustrated in figure 47.

```
Goal:
  isa PRS
  lag2 Paper
  lag1 Rock
  lag0 nil
```

*Figure 47.* An example ACT-R chunk.

When the model's opponent makes a move, a chunk encoding its last three moves is put into the goal, and then popped to make it a chunk in memory (either creating a new chunk or reinforcing an existing chunk).

To predict the opponent's move, the model attempts to retrieve a chunk from memory that matches the opponent's last two moves (slots lag1 and lag2). The value of

the lag0 slot is the move the model predicts its opponent to make, and so plays the move that beats it.

In Lebiere & West (1999), both humans and the lag 2 ACT-R model were pitted against lag 1 and lag 2 versions of the ACT-R model. Consistent with the findings in West (1998), both humans and the lag 2 ACT-R model were able to beat the lag 1 opponent; however, exact scores were not reported.

Several ACT-R models of human Rock=2 PRS play were presented in Rutledge-Taylor & West (2004). These models were similar to those appearing in Lebiere & West (1999), however, they differed in that they were designed to be sensitive to the unequal payoffs in the Rock=2 game. This sensitivity was achieved by reinforcing certain kinds of chunks more than others depending on what the opponent's last play was. This was done by harvesting these chunks twice. Rutledge-Taylor & West (2005) tested three variations on this strategy. One model paid extra attention to cases when its opponent played rock; another attended more closely to scissors; while the third attended more closely to both rock and scissors (effectively paying less attention to paper).

The result was that the third model provided the best match to the human data. This makes intuitive sense in that a human player is likely to incorporate a defensive component to his or her game, which is to be wary of when the opponent is likely to play Rock; however, he or she might also incorporate an offensive component which is to also focus on when the opponent might play scissors. Winning with Paper, or losing to a Paper play, is a less important event in the game.

### DSHM PRS Models

Given the broad similarities between DSHM and the declarative memory system of ACT-R (Rutledge-Taylor & West, 2008), the DSHM models here were based on the ACT-R models described above.

The DSHM models took sequences of opponent's plays, encoded as ordered complex items as input. The items consisted of two, three, or four atomic items, for lag 1, lag 2 and lag 3 models respectively; the extra item is the predicted or current play by the opponent. The right most item represented the opponent's last move, while items to the left represented previous plays. For example, if the opponent's last few plays were "rock, paper, paper, rock, scissors", a lag 1 DSHM model would learn the following pattern [rock:scissors], after scissors was played, reinforcing the association between 'scissors' as a play the follows 'rock'. A lag 3 DSHM model would learn the pattern [paper:paper:rock:scissors], reinforcing the sequence of the last four plays.

An interesting architectural point that should be made here is that DSHM reinforces all of the combinations of consecutive sets of sub-items in the input, when learning ordered complex items. Thus, given the lag 3 input above, the following sequences of items are reinforced after scissors is played: [paper:paper:rock:scissors], [paper:rock:scissors], [paper:paper:rock], [rock:scissors], [paper:rock], and [paper:paper].

From this is apparent that DSHM models of two or more lags incorporate some of the learning of shorter lagged models as well. An additional consideration is that this results in a potential liability for DSHM, as shorter sequences receive repeated reinforcement for several consecutive trials. In this example, this will have been the third time that [paper:paper] had been reinforced: the third time just now; the second time,

when ‘rock’ was played; and the first time when the most recent of the two ‘paper’ plays was made. It is also the second time that [paper:paper:rock] will have been reinforced (the first time being when ‘rock’ was played). Thus, in this respect, DSHM models of PRS differ somewhat from both the perceptron-like networks, and the ACT-R models discussed above.

#### *Rock=1 Simulations*

A variety of DSHM PRS players were built. The manipulated parameters were: number of lags and the length of the vectors used to represent items. It would not be appropriate to embark on a complete discussion of the inner-workings of DSHM here (see *Dynamically Structured Holographic Memory: The Mechanisms*, and *Dynamically Structured Holographic Memory: Cognitive Interpretation*). For now, it is sufficient to understand that vector length is correlated with memory capacity. Additionally, lower vector lengths contribute to noise-like effects due to an increased amount of interference in the system.

Each combination of two sets of values, listed in table 23, were used to build a unique DSHM PRS player.

*Table 23. Parameter values tested*

Parameter	Values
Lags	1, 2, 3
Vector Length	32, 64, 128, 256, 512, 1024, 2048

Each DSHM model played against the aggressive lag 1, aggressive lag 2, and passive lag 2 network models.

### *Evaluation*

To determine which DSHM performed most like human players, data from West & Lebiere (2001) was used as a comparison: human players average 9.99 (s.d. 19.61) more wins than the aggressive lag 1 networks, after 300 trials; lost to the aggressive lag 2 models by an average margin of 8.89 (s.d. 19.74), after an imprecise number of trials; and, beat the passive lag 2 by 11.14 wins after 287 trials.

Given that understanding human play against the aggressive lag 2 networks is difficult due confounding factors discussed in West & Lebiere (2001), and the fact that an exact target win difference (after a fixed number of trials) is not available, comparison to the data against this opponent was simplified.

The DSHM models were rated according to the mean squared difference between their average final scores against the aggressive lag 1 and passive lag 2 networks, and the average final scores of humans against these models. Additionally, DSHM models that win against the aggressive lag 2 were disqualified as potential models of human play. This is because all that is certain about human performance against the aggressive lag 2 networks is that humans lost to these networks, on average. Additionally, West & Lebiere (2001) discuss factors that could make the interpretation of human players' performance against the aggressive lag 2 networks difficult.

### *Results*

Of all the models tested, one produced results that came very close to the human data. The Lag 3 DSHM model with vector lengths of 1024 scored an average of 10.89

wins more than the aggressive lag 1 network, 13.24 more wins than the passive lag 2, and lost to the aggressive lag 2 by an average of 6.22 wins per game.

The fact that the best DSHM model was a lag 3, not a lag 2 model, was surprising at first. Lebiere & West (1999), and West & Lebiere (2001), found that lag 2 ACT-R and lag 2 network models provided the best fit to the human data. However, the fact that the DSHM lag 3 models incorporated lag 2 and lag 1 memory behaviour makes this result more consistent with previous findings. The DSHM lag 3 model weighs lag 1 sequences the most, lag 2 sequences second, and lag 3 sequences the least. So, it could be argued that, on average, the lag 3 DSHM models are more like lag 2 ACT-R and network models, than the lag 3 ACT-R and network models.

#### *Rock=1 Model Comparison*

This paper reviews the three different types of models of PRS play: ACT-R, DSHM, and perceptron-like networks. In each case, the model of human play, played games consisting of 300 trials against the aggressive lag 1 network, and games of 287 trials against the passive lag 2 network. For each model, the mean difference in the number of wins scored by the model and the opponent network was recorded. Human scored, on average, 9.99 more wins than the aggressive lag 1 networks, and 11.14 more wins than the passive lag 2 network. The sum of the squares of the differences between the model's results and the human results are presented as a basis for comparing the model's fit to the human data.

The best ACT-R model was taken from Rutledge-Taylor & West (2005). It was exemplar based, and used the following parameters: ANS=0.28, OL=NIL. The best

DSHM model was the lag 3, vector length 1024, model discussed above. The best network model was the aggressive lag 2 network.

New network versus network simulations were run for this comparison: 10000 games were run against each of the two benchmark opponents. The mean difference in wins between the aggressive lag 2 and aggressive lag 1 networks was somewhat lower than was found in West & Lebiere (2001). However, given the high standard deviation on the win differences, both the results found here and those found in West & Lebiere (2001) may be valid.

Table 24 summarizes the best models of human Rock=1 PRS play. The DSHM produces the closest fit to the human data, i.e., it scored the lowest mean squared error. Additionally, the DSHM model lost to the aggressive lag 2 model by a mean win difference of 6.22, which help support this model as a good account of human PRS play.

*Table 24. Models of human rock=1 PRS*

Opponent	Human	ACT-R	DSHM	Network
Agg. lag 1	9.99	12.30	10.89	5.76
Pas. lag 2	11.14	8.15	13.24	10.44
Rating		14.27	5.22	18.37

#### *Rock=2 Simulations*

The mechanism for building a sensitivity to the unequal payoffs of the Rock=2 game in the DSHM models was essentially the same as for the ACT-R Rock=2 models.

Three versions of the Rock=2 DSHM models were created: one that attended to opponents' rock plays more; another that attended to scissors more; and, one that gave preference to both rock and scissors. This extra attention was achieved by training the models twice on sequences ending in these plays.

For each of the three variations on the Rock=2 DSHM player, the combinations of the three different lags and seven vector lengths were tested.

### *Evaluation*

Each of the DSHM players results were compared to the human data from Rutledge-Taylor & West (2004). A mean squared error approach was used. Six data point were compared: the mean points differences versus the three network models, and the strategy indices versus these opponents. Because the strategy index values were, on average smaller than the point differences, and because they are crucially important to establishing strategic play, these three data points were given twice the weight of the points difference comparisons. Thus, each model's rating was the sum of the squares of the mean point differences and twice the sum of the squares of the strategy indices.

### *Results*

The results were somewhat predictable based on the DSHM rock=2 and ACT-R Rock=2 results: The lag 3 DSHM model with vector lengths of 1024, and that paid extra attention to both rock and scissors produced the best fit to the human data. It matched five of the six data points very well. However, this model failed to defeat the aggressive lag 2 network model. There were other DSHM models that beat the aggressive lag 2 network, but failed to match the other five data points well (e.g., the margins of victory were too great).

*Rock=2 Model Comparison*

Three different kinds of models of human Rock=2 PRS play are discussed in this paper: ACT-R, DSHM and perceptron-like networks. The ACT-R results are taken from Rutledge-Taylor & West (2004), the DSHM model is the one discussed above. As with the Rock=1 comparison, new network versus network data was collected. The network model of human Rock=2 played each of the benchmark opponents 10000 times. Each model played 300 trial games against the three network opponents discussed in Rutledge-Taylor & West (2004). The evaluation method for all types of models was the same as for the DSHM models discussed above.

Figures 48 and 49 summarize the evaluations of the three model types. The confidence values for the human data in figure 48 are estimated based on the 95% confidence intervals of a regression analysis of the rate of point difference achieved by the human players against each of the three opponents.

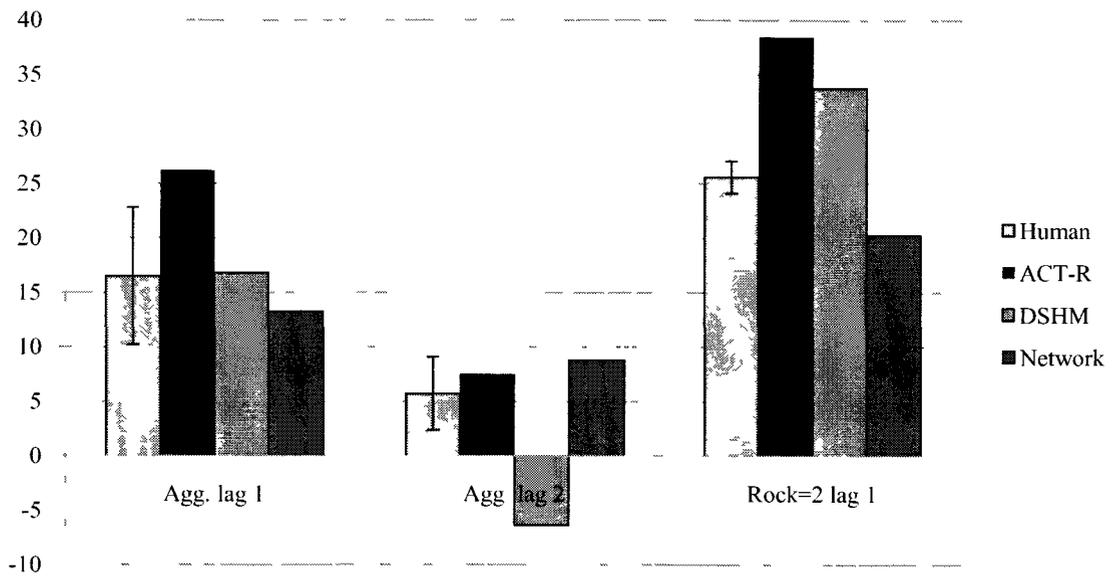


Figure 48. Points difference comparison for Rock=2 PRS. Human values include estimated 95% confidence values.

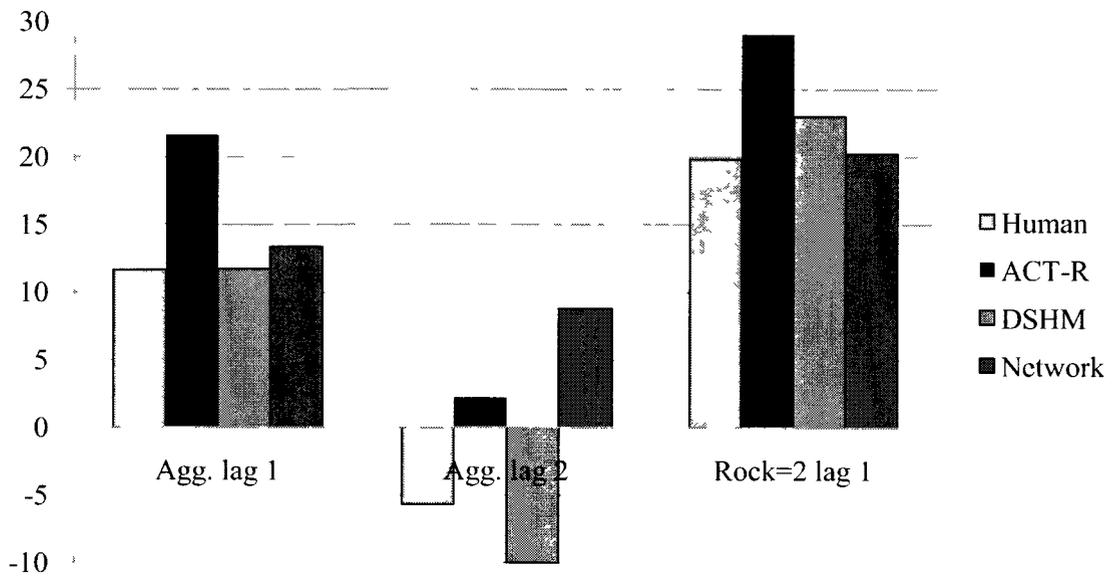


Figure 49. Strategy indices comparison for Rock=2 PRS.

All three models produced good fits to the human data. However, the DSHM model is unique in that it failed to beat the aggressive lag 2 network. However, it correctly scored a negative strategy index versus this opponent. In contrast, the ACT-R and network models beat the aggressive lag 2, but did not score negative strategy indices, as did the human players. Thus, there is an obvious objective in building a superior model of human play. That is, to build a model that beats the aggressive lag 2, but does so despite a negative strategy index.

### Conclusions

The simulations run and analyzed here demonstrate that DSHM can, in fact, be used successfully to model at least one memory task that relies on reconciling inconsistent information and rapidly changing predictions based on past events. Despite the fact that DSHM was designed to model only long-term memory, it may also be useful as a model of short-term memory. This also suggests that long-term and short term memory in humans may rely on the same basic mechanisms.

### References

- Lebiere, C. & West, R. L. (1999) A dynamic ACT-R model of simple games. In *Proceedings of the 21<sup>st</sup> Annual Conference of the Cognitive Science Society*. 296-301. Simon Fraser University: Vancouver, Canada.
- Rutledge-Taylor, M. F., Pyke, A. A., West, R. L. & Lang, H. (submitted) Modeling a three term fan effect. In *Proceedings of the Tenth International Conference on Cognitive Modeling*. Philadelphia, PA: Drexel University.

- Rutledge-Taylor, M. F., Vellino, A. & West, R. L. (2008) A holographic associative memory recommender system. In *Proceedings of the Third International Conference on Digital Information Management*. 87-92. London, UK.
- Rutledge-Taylor, M. F. & West, R. L. (2004) Cognitive modeling versus game theory: Why cognition matters. In *Proceedings of the Sixth International Conference on Cognitive Modeling*, 255-260. Pittsburgh, PA: Carnegie Mellon University/University of Pittsburgh.
- Rutledge-Taylor, M. F. & West, R. L. (2008) Modeling The fan effect using dynamically structured holographic memory. In *Proceedings of the 30<sup>th</sup> Annual Conference of the Cognitive Science Society*. 385-390. Washington, DC.
- West, R. L. (1998) Zero sum games as distributed cognitive systems. In *Proceedings of the Complex Games Workshop*. Tsukuba, Japan: Electrotechnical Laboratory Machine Inference Group.
- VonNeumann, J., & Morgenstern, O. (1944) *Theory of Games and Economic Behaviour*. Princeton, NJ: Princeton University Press.

## APPENDIX

The papers in this section do not make central contributions to the theoretical and practical claims made in this dissertation. However, they both support the main portfolio papers presented above, and are not widely available. Thus, they are made available here for the reader's convenience

## INTERFERENCE AND ACT-R: NEW EVIDENCE FROM THE FAN EFFECT

Previously published as West, R. L., Pyke, A. A., Rutledge-Taylor, M. F. & Lang, H. (2010). Interference and ACT-R: New evidence from the fan effect. In D. D. Salvucci & G. Gunzelmann (Eds.), *Proceedings of the Tenth International Conference on Cognitive Modeling* (pp. 277-281). Philadelphia, PA: Drexel University.

### Contribution

An important finding presented in this paper is the fact that the fans of all the terms in a sentence affect recognition reaction time of the sentence. ACT-R does not predict this; however, DSHM does. A DSHM model of this experiment is discussed in Rutledge-Taylor, Pyke, West and Lang (2010), which is included in this portfolio.

This finding supports the priority DSHM places on interference as the main feature of memory responsible for accounting for many memory phenomena (See *Dynamically Structured Holographic Memory: Cognitive Interpretation*).

### Abstract

We present data demonstrating that interference plays a role in the fan effect. We also show that this cannot be accounted for using ACT-R. An ACT-R model is fit to the data and we discuss options for altering the model to account for the data.

### Introduction

The fan effect refers to the fact that cues that are associated with more facts result in slower recall than cues that are associated with fewer facts. For example, in the study

that established the fan effect, Anderson (1974) asked subjects to memorize facts about where various different characters had been seen. Subjects were then shown a cue with a character and a place and asked if it was true (i.e., if they occurred together in the set of facts subjects had memorized). Overall, the more places a character had been, the slower subjects were to confirm that it was either true or false. Also, subjects were slower to say false than they were to say true.

In the ACT-R architecture (Anderson & Lebiere 1998) the cue is held in a buffer as a chunk and each slot value of the cue spreads activation to chunks in declarative memory that have the same slot values. For example, if the cue chunk is person:hippy location:park, then hippy will spread activation to all chunks that have hippy as a slot value and park will spread activation to all chunks that have park as a slot value (note, the slot names do not play a role). The number of lines of activation leaving from a slot value in the cue is the fan of that slot value, and the fan of the cue is the sum of the fans of its slot values.

In ACT-R, the amount of activation spread from a cue to a chunk is theorized to be proportional to the number of past associations between slot values of the cue and the chunk. In the ACT-R architecture, the way of calculating this is based on an assumption that exposure to different facts has been counterbalanced, as in a psychology experiment (Anderson & Reder, 1999). If it is assumed that everything has been counterbalanced and the number of exposures per chunk is equal then the activation can just be divided evenly among the chunks. So, the higher the fan the lower the amount of activation delivered to each individual chunk (see Anderson & Reder, 1999, for how to use ACT-R when exposures have not been counterbalanced). Anderson and Reder (1999)

modeled the fan effect in ACT-R by assuming that people retrieve the most active chunk and respond true (i.e., they have seen it before) if the retrieved chunk matches the cue, and false (i.e., they haven't seen it before) if it does not.

One consequence of this model is that only the spreading activation received by the chunk that is chosen affects the reaction time (RT). In other words, there is no interference from the activation of other chunks. However, as fan goes up so do the number of other chunks that receive activation. As part of a fan experiment we tested the effect of these "other" chunks to see if interference plays a role in the fan effect and how that might alter the ACT-R fan model.

### Experimental Design

In our experiment we created false cues by taking a true fact and replacing one element with a false element. For example, if subjects had studied the fact that the red hat is in the kitchen, we could create a false cue by replacing hat with pen. Under these conditions the ACT-R model predicts that the fan of the false element of the cue will have no effect on retrieval time, unless the original fact is not retrieved. However, we performed simulations with the ACT-R fan model and found that in our experimental design, the chunk representing the original version of the fact always received the most activation, and therefore was always chosen (as far as we can see this is also true for other fan experiment designs, but it is possible to create more extreme differences in fan where this would not be true). Related to this, the fan of the false element should also have no effect on the error rates. Although the ACT-R fan model does not explicitly model errors, errors would be due to noise and the retrieval threshold. This could

conceivably interact with fan for the chunk that is being retrieved but the fan of the false element does not affect this chunk.

## Method

### *Subjects*

Twenty seven participants (11 males and 16 females: mean age 19.9 years, SD = 2.2) were recruited from introductory psychology courses at Carleton University to take part in the experiment. Participants received course credit as compensation for their time.

### *Procedure*

The experiment was divided into three main phases: A study phase, a recall phase and a recognition phase. In the study phase each participant was assigned one of three unique sets of study sentences and was instructed to memorize the sentences in the list. Once the participant indicated that they were prepared to proceed, the recall portion of the experiment began.

The study set consisted of sixteen sentences of the form, “The color thing is in the place”. The color term was one of ten colors; the thing was one of ten house-hold items; and the place was one of ten locations in/around a typical home. Very typical item/locations combinations, such as ‘comb’/‘bathroom’, were not used in generating the study set sentences. Each term could have a fan of either 1 or 4. Thus, the four possible sentence fans were: 3, 6, 9, and 12.

In the recall phase each participant was tested three times. Each test began with the participant trading the study set with the experimenter for a new list of sentences identical to the study set, but with one term from each sentence replaced with a blank,

and the order of the sentences randomized. The participant's task was to correctly fill-in each of the blanks with the missing word. The participant was given as much time as he or she needed.

Once finished, the experimenter recorded the number of correct responses and for each error, provided the correct missing word to the participant. The participant was then given the opportunity to review the study set before being tested again. The three tests were balanced such that each term from each sentence in the study set was replaced with a blank exactly once. After the third iteration the recognition phase began.

The recognition phase of the experiment was conducted on a computer using Experiment Builder (by SR Research). The participant's task was to correctly judge whether sentences presented in the middle of a 17" CRT display were members of the study set, or not. Accuracy and reaction time data were recorded for each trial. Each participant was presented with 96 test sentences, which consisted of three exposures to each of the study set sentences, and 48 sentences that were not from the study set.

The participants were told that they should consider sentences from the study set to be true, while all others should be considered false. Each false sentence was generated by swapping one of the three terms from a true sentence with another term from the same category (e.g., color, thing, or place) and with the same fan. Each true sentence was used to generate three different false sentences. Thus, for each exposure to a true sentence there was a false sentence with the identical fan. Once the test sentence appeared the participant would indicate if the sentence was in the study set by hitting the 'z' key, or if it was not by hitting the '/' key.

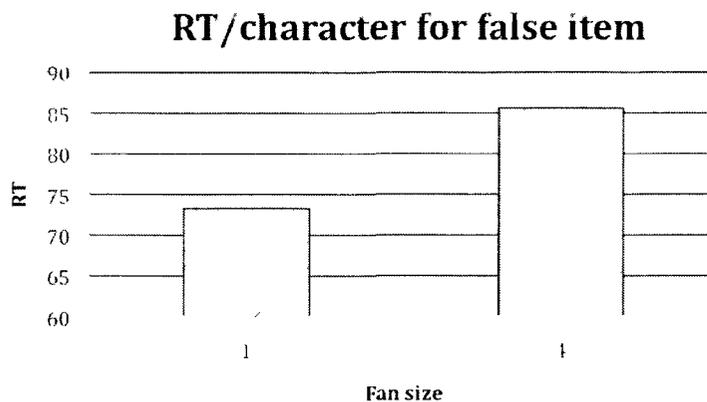
## Results

The data from one of the participants was excluded from the results presented below. This was due to the fact that this participant's performance was significantly poorer than all other participants by a large margin ( $P < 0.001$ ). The results below reflect the data collected from the remaining 27 participants. By the end of the third iteration of the recall phase the participants were correctly completing the sentences 91.4 percent of the time. The results of the recognition phase replicated the fan effect. These results are reported in Rutledge-Taylor, Pyke, West, & Lang (2010). However, in this paper we will focus on the results related to the predictions described above and fitting an ACT-R model to the data.

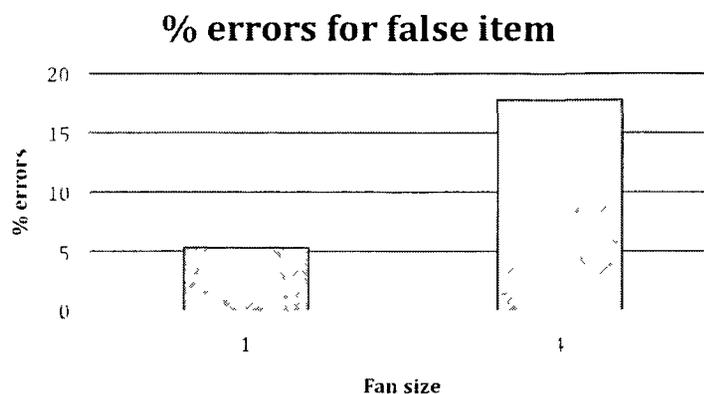
The hallmark of a good scientific theory is that it makes precise, falsifiable predictions. Many theories in Psychology fail to meet this criterion. However, because ACT-R is precisely specified, models built in ACT-R are more readily falsifiable. To test the predictions of the ACT-R fan model (Anderson & Reder, 1999) concerning the fan of the false items we ran an ANOVA testing for the effect of the fan of the false items on RT and error rate. RT was significantly higher when the fan of the false item was equal to 4 than when it was equal to 1 ( $P < 0.001$ ). The error rate was also significantly higher when the fan of the false item was equal to 4 than when it was equal to 1 ( $P < 0.001$ ). We also ran a correlation between the fan of the false items and RT, with the fans of the true items partialled out. We found a significant correlation of  $r = 0.156$  ( $P < 0.001$ , one tailed). Similarly, we ran a correlation between the fan of the false items and % errors, with the fans of the true items partialled out. Here we also found a significant correlation of

$r=0.193$  ( $P<0.001$ , one tailed). The size of these correlations was roughly similar to the same correlations done with the fan of the true items.

Contrary to the predictions of the ACT-R fan model, we found that the fan of the false items significantly affected RT such that a larger fan led to slower responses (see Figure 50). Consistent with this and also contrary to the predictions of the model, we found that the fan of the false element significantly affected the error rate such that a larger fan led to more errors (see Figure 51). These results indicate that interference from the false item plays a role in the fan effect.



*Figure 50.* Reaction time in msec/character for responding false to a false cue as a function of the fan of the false item in the cue.



*Figure 51.* Percent errors for responding false to a false cue as a function of the fan of the false item in the cue.

#### *Model Evaluation*

Although falsification of a model is sometimes viewed as a bad thing, falsification actually shows that a model was well specified. Falsification also creates an opportunity to move toward a better model. To this end we fit the ACT-R fan model to our data. Anderson and Reder (1999) used the following function,  $RT = I + Fe^{-Ai}$ , to calculate RT where F is a scaling constant for time, I is a constant representing how long it takes subjects to make their response after they know it, e is the base for natural logarithms and A is the activation of the chunk (which includes spreading activation). Activation was calculated as,  $A = B + S$ , where B is base level activation and S is spreading activation. We fitted the Anderson and Reder (1999) ACT-R fan model to our data using identical parameter values, except that we had to increase S slightly from 1.45 to 2 to avoid getting negative activation values. As in Anderson and Reder (1999), B was set to zero because it trades off with S.

We eventually figured out that the slight difference in  $S$  was because we used the current method of calculating fan size in ACT-R 6, which is to add 1 to the fan of each item in the cue to represent the match between that item and a chunk in memory representing that item. For example, 1 would be added to the fan of cup because it is assumed that we all have a chunk in declarative memory representing cup. In contrast, Anderson and Reder (1999) calculated the results without adding 1 to fans of the items in the cue. Whether or not to do this is an interesting issue. However, we recalculated our results without adding 1 and found it made very little difference to our results or conclusions.

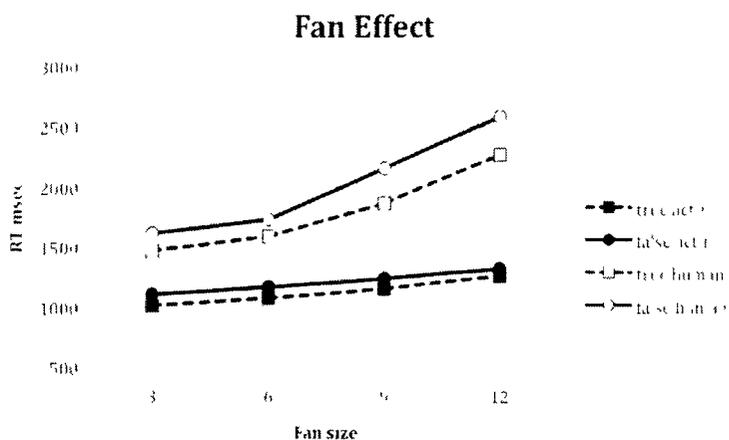
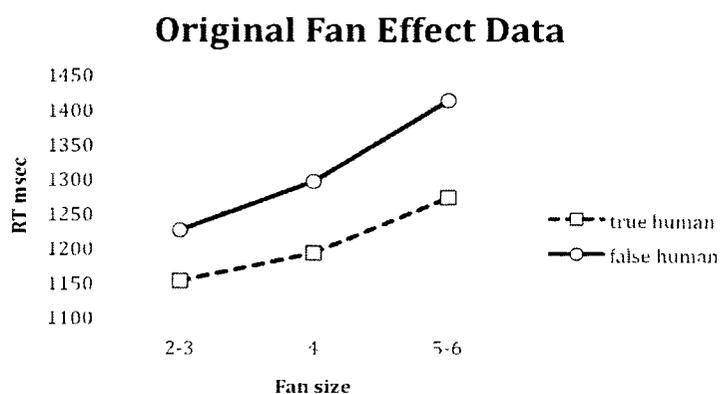


Figure 52. The original Anderson & Reder (1999) ACT-R.

Figure 52 shows the fit of the original ACT-R fan model to our data. The fact that the model predicts an overall lower RT is not significant as it can be accounted for by assuming our subjects took longer to press the true/false keys, which can be modeled by increasing the  $I$  parameter. However, the shape of the functions and the relationship

between the functions is clearly different. The human data shows a clear upward curve that the model does not and the model RTs converge as fan increases while the human data diverges.

Figure 53 shows the original fan effect data from Anderson and Reder (1999) re-plotted. Note that it shows the same divergence and upward curve. In fact, the original ACT-R model for this data (faithfully recreated and shown in Figure 54) also shows a slight upward curve for the true cues, but not for the false cues. Also, as with our data, the false function diverges from the true function as fan goes up. However, it is important to keep in mind the scale of the graphs and realize that these effects are much smaller in the Anderson and Reder (1999) data and may not even be real, although, the consistency of this result across conditions and studies indicates that we should take it seriously.



*Figure 53.* Re-plotted data from Anderson and Reder (1999).

*An Alternative Model*

Next we addressed the issue of the parameter values. Specifically, we wanted to know if the ACT-R model could be made to fit the data. The only way that we could find to fit the data was to use the latency exponent parameter ( $f$ ) that is available in ACT-R 6. This parameter, which has rarely been used, changes the RT function to,  $RT = I + Fe^{-fA}$ . By setting  $f=3$  and increasing  $F$  from 613 to 2000 we obtained a good fit to the data (see Figure 54 - note, that the  $I$  parameter could be increased to overlap the functions but it is easier to see this way). Increasing  $f$  lowers overall RT, so increasing  $F$  can be viewed as a way of compensating for this. The other effect of raising  $f$  was to increase the acceleration of the rate at which lowering activation raised RT. We will refer to this as the ACT-R( $f$ ) model (see Figure 55). However, please note that this model violates the ACT-R modeling convention of using established parameter values unless you have a justification (Anderson & Lebiere, 1998).

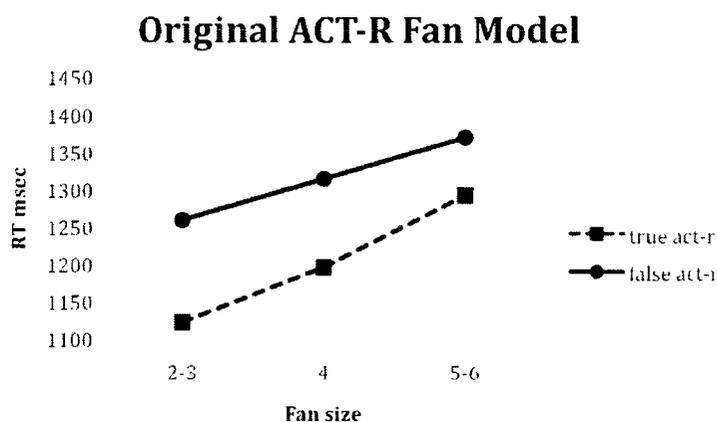
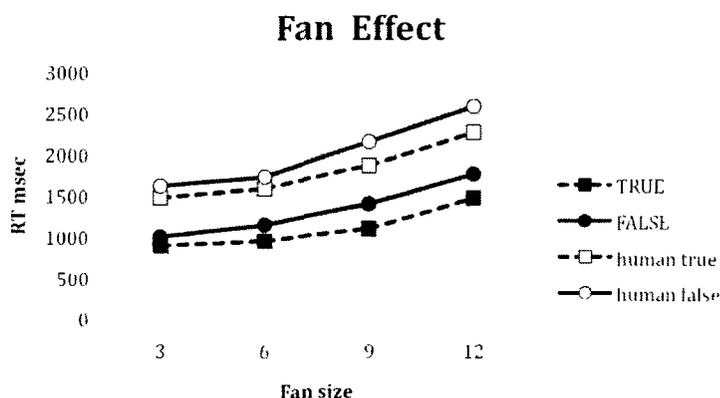


Figure 54. A re-creation of the ACT-R fan model from.

*Rationalizing the alternative model*

There are three ways we can interpret the ACT-R(f) fan model. We know that it cannot account for our finding that the fan of the false item in the cue affects RT and % error any better than the normal ACT-R fan model. However, it is possible to interpret the manipulation of  $f$  as representing the aggregate effect of interference. In this case,  $f$  would be related to the total effect of interference in the task. If we assume that our use of more cue items and higher fans produced greater overall levels of interference, then the fact that our results show a more pronounced nonlinear effect than the Anderson and Reder (1999) results could be modeled by increasing  $f$  to represent higher levels of interference. In this sense, ACT-R could be adjusted to account for the presence of interference but could not be said to include a (process) model of interference. More studies would be required to see if  $f$  actually does function this way.



*Figure 55.* The ACT-R ( $f$ ) model fit to our data (note, that the  $l$  parameter could be increased to overlap the functions).

A less charitable approach to understanding the ACT-R(f) model would be to point out that adding  $f$  to a model that already has a lot of parameters creates a system capable of fitting a lot of different functions. We had no principled reason to adjust  $f$  and found that it worked as part of a parameter tweaking process that involved all of the available parameters. So possibly the fit of this model is merely fortuitous.

A third, more constructive way of viewing it is to see the manipulation of  $f$  as a proxy for an additional mechanism or process - in this case, interference. Although ACT-R does not include an interference mechanism, modifications have been introduced to do this. For example, the spacing effect modification of Pavlik and Anderson (2005) assumes that interference plays a role in order to account for the spacing effect in memory. Similarly, the semantic interference modification proposed by Van Maanen & Van Rijn (2007) assumes a form of interference to account for the Stroop effect. Likewise, our findings indicate the need for an explicit model of interference in ACT-R. A simple way of doing this that is consistent with our manipulation of  $f$  is to introduce a penalty that reduces activation based on the total fan of the information in the cue - the higher the overall fan, the greater the penalty for all chunks receiving spreading activation. We could create such a function but it would not be meaningful at this point since it would be custom made to fit our data. Essentially, this would have the same effect as raising  $f$ , but the effect would be tied to the overall fan and therefore would account for our finding that the fan of the false item in the cue affects RT and % error.

### Model Re-Evaluation

To gain further insight into the ACT-R (f) model we applied those parameter settings to our recreation of the Anderson and Reder (1999) fan model. The results are illustrated in Figure 56. The true results actually produce a reasonable fit to the data but the false results clearly do not fit. This could be because the fit of the ACT-R (f) model to our data was merely fortuitous, or it could be because higher  $f$  values are only appropriate when interference is higher, as suggested above.

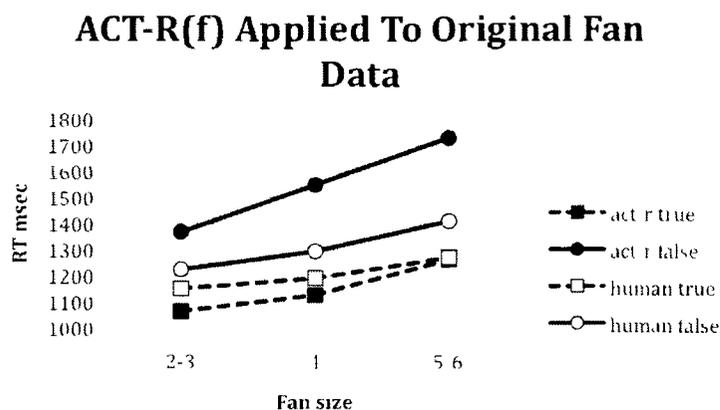
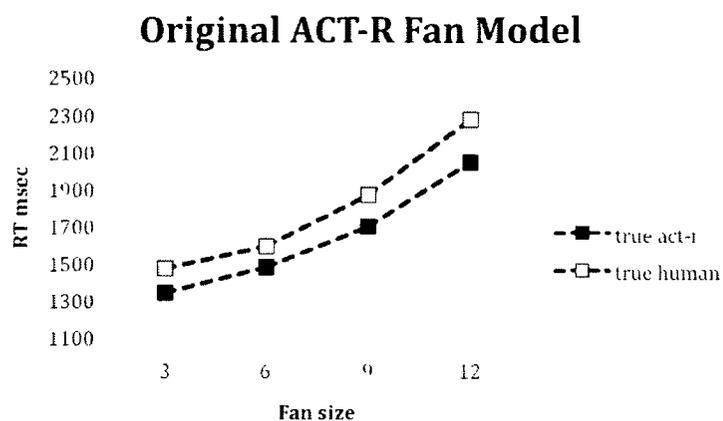


Figure 56. The ACT-R (f) model applied to the data from Anderson and Reder (1999).

Based on our experimental findings showing that the ACT-R fan model for correctly identifying false cues cannot be correct, we also tried fitting Anderson and Reder's (1999) fan model to our data for the true results only (see Figure 57). Without having to fit the false data we were able to get a good fit by adjusting only  $F$  and  $I$  ( $F=1000$ ;  $I=1100$ ;  $S=1.45$ ; similar to Anderson and Reder we did not add  $l$  when calculating the fan). This is much less problematic because it avoids adjusting  $f$ , which is almost never altered in ACT-R modeling. Also, it is important to remember that there is

variability associated with the human data so it is likely that a single intermediate value of  $F$  could be used to obtain a reasonable fit to our data and Anderson and Reder's (1999) data.



*Figure 57.* The Anderson and Reder (1999) ACT-R fan model fit to our data for correctly identifying true cues only.

### Conclusions

Our results show that the ACT-R fan model for correctly identifying false cues cannot be completely correct. Also, although fitting ACT-R to our data was possible, it was also problematic because it required unprecedented alterations to the parameter values as well as assumptions about the meaning of those alterations that remain untested. However, when we did not try to fit the ACT-R fan model for correctly identifying false cues, the ACT-R fan model for correctly identifying true cues fit our data well, without any problematic parameter alterations. Based on this, it appears most likely that the

problem lies with the assumptions and processes behind the ACT-R fan model for correctly identifying false cues.

#### References

- Anderson, J. R. (1974). Retrieval of propositional information from long-term memory. *Cognitive Psychology*, 6, 451-474.
- Anderson, J. R., & Lebiere, C. (Eds.) (1998). *The atomic components of thought*. Mahwah, NJ: Lawrence Erlbaum.
- Anderson, J. R. & Reder, L. R. (1999). The fan effect: New results and new theories. *Journal of Experimental Psychology: General*, 128(2), 186-197.
- Pavlik, P. I., Jr., & Anderson, J. R. (2005). Practice and forgetting effects on vocabulary memory: An activation-based model of the spacing effect. *Cognitive Science*, 29, 559-586.
- Rutledge-Taylor, M. F., Pyke, A. A., West, R. L. & Lang, H. (2010). Modeling a three term fan effect. In D. D. Salvucci & G. Gunzelmann (Eds.), *Proceedings of the Tenth International Conference on Cognitive Modeling* (pp. 211-216). Philadelphia, PA: Drexel University.
- Van Maanen, L., & Van Rijn, H. (2007). An accumulator model of semantic interference. *Cognitive Systems Research*, 8(3), 174-181.

# COGNITIVE MODELING VERSUS GAME THEORY: WHY COGNITION MATTERS

Previously published as Rutledge-Taylor, M. F. & West, R. L. (2004). Cognitive modeling versus game theory: Why cognition matters. In M. Lovett, C. Schunn, C. Lebiere & P. Munro (Eds.), *Proceedings of the Sixth International Conference on Cognitive Modeling* (pp. 255-260). Pittsburgh, PA: Carnegie Mellon University/University of Pittsburgh.

## Contribution

This paper reports on the original Rock=2 Paper, Rock, Scissors experiments. These results are referenced in Using DSHM to Model Paper, Rock, Scissors. In this paper only a neural network model of human play is presented.

## Abstract

We call into question game theory, as an account of how people play two player zero-sum games. Evidence from a modified version of the game Paper, Rock, Scissors suggests that people do not play randomly, and not according to certain play probabilities. We investigated the relationship between game theory predictions and a cognitive model of game playing based on the detection of sequential dependencies. Previous research has shown that the sequential dependency model can account for a number of empirical findings that game theory cannot. The sequential dependency model has been implemented using both simple neural networks and ACT-R. In this paper we used

simple neural networks (a description of how our findings relate to the ACT-R model is included in the Conclusion section). For simple games, such as Paper, Rock, Scissors, game theory has been able to correctly predict aggregate move probabilities. In this paper we show that this is an artifact of the symmetry of the payoffs, and that for asymmetrical payoffs the game theory solution does not predict human behavior. Furthermore, we show that the model of game playing that underlies game theory cannot be used to predict the results no matter what move probabilities are used. Finally, we show that the results can be accounted for by augmenting the network sequential dependency model so that the reward system is related to the game payoffs.

### Introduction

Game theory (VonNeumann & Morgenstern, 1944) was not created as a cognitive model. That is, it was not intended to account for mechanisms underlying the manner in which humans play games. Rather, it was intended to be a formal mathematical system for understanding game playing from a rational perspective. However, game theory has had an enormous influence on theories and ideas about how people play games. Because empirical studies have shown that game theory is poor at predicting how humans play games, most researchers do not believe that game theory is a good model of how humans process information during games (Pool, 1995). Nevertheless, the game theory model of a player is still extremely influential in terms of how we understand human game playing.

Game theory is a large collection of mathematical principles that are used in accounts of a variety of human interactions. It is used by a variety of researchers including economists, mathematicians, and behavioral psychologists to name a few. The

focus of this paper is specific to the application of game theory as an account of how people play a relatively constrained set of non-zero sum games. Henceforth, the use of the term 'game theory' will be used to refer to a subset of principles within game theory commonly used to describe this type of game.

In game theory, a game is defined as a situation in which two players each select one move from a discrete set of choices, the combination of which determines a payoff for each player. There are two varieties of games. In zero-sum games the payoffs for the players must sum to zero, and there is no option for cooperation (i.e., one player's gain is always equal to the other player's loss). In non-zero-sum games this constraint does not exist and cooperation can be an option. In either case, to apply game theory it is necessary to assume that players can make random selections from their move choices according to specific probabilities assigned to each move. This amounts to two assumptions about the cognitive abilities of players: (1) they have some way of calculating or learning move probabilities, and (2) they are able to select moves at random according to these probabilities. In what follows, we will refer to these assumptions concerning players' abilities as *the game theory player model*. Assuming that a player matches the game theory player model, it is generally possible to use game theory to calculate the optimal set of move probabilities. We will refer to the outcome of game theory calculations as *the game theory solution*.

Most research on human game playing has focused on non-zero-sum games, where it has been shown that people generally do not follow the game theory solution. One reason for this is that people are often influenced by factors that are extraneous to the game theory solution, such as social norms concerning cooperation (Poundstone, 1992;

Samuelson, 1997). However, these results show only that people do not play according to the game theory solution. The game theory player model is still viable as it is possible to explain the results in terms of people playing according to the game theory player model (i.e., probabilistically), but with move probabilities inconsistent with the calculated game theory solution. In zero-sum games there is no option to cooperate so there should be less extraneous influences. However, there is very little direct research on human zero-sum game playing (Poundstone, 1992). The general view of human zero-sum game playing is based mainly on experimental comparisons between humans and the game theory player model. With regard to this, psychological studies have shown that people are very poor at the two essential skills required by the game theory player model: (1) learning optimal move probabilities (e.g., Gazzaniga,

1998), and (2) behaving randomly (see Wagenaar, 1972 for a review).

Based on this, a common view of individual humans as game players is that we are poor game theory players. That is, we play in a way consistent with the game theory player model but we are not good at learning the optimal move probabilities, we are poor at being random, and we are influenced by considerations extraneous to the game theory framework (e.g., norms about cooperation). Attempts have been made to adapt the game theory player model to better capture human behavior. The focus in this area has been on introducing a learning component to explain how humans acquire move probabilities. Game theorists have studied the effects of using learning algorithms to acquire move probabilities (Fudenberg & Levine, 1998), and there have been some attempts to build cognitively plausible versions of the game theory player model using cognitive architectures (e.g., see Ritter & Wallach, 1998 for examples using ACT-R and SOAR).

However, all of this work is based on the implicit acceptance of the game theory player model as an appropriate framework for understanding human game playing behavior.

### An Alternative Player Model

A psychologically plausible alternative to the game theory player model is that instead of trying to learn advantageous move probabilities, people try to detect sequential dependencies in their opponents' play and use this to predict their opponents' moves (Lebiere & West, 1999; West, 1998; West & Lebiere, 2001). That is, players learn recognize patterns of consecutive moves in their opponents' play. This model is consistent with a large amount of psychological research showing that when sequential dependencies exist, people can often detect and exploit them (e.g., Estes, 1972; Restle, 1966; Rose & Vitz, 1966; Vitz & Todd, 1967). It also explains why people tend to do poorly on tasks that are truly random; they persist in trying to predict the outcomes even though doing so results in sub-optimal results (e.g., Gazzaniga, 1998; Ward, 1973; Ward, Livingston, & Li, 1988).

West and Lebiere (2001) used neural networks to examine the possibility that people play games by attempting to detect and exploit sequential dependencies in their opponent's play. The networks were designed to detect sequential dependencies in the game of Paper, Rock, Scissors (hence forth PRS). PRS was chosen because it is familiar to most people and because it is very easy to play. It is also a zero-sum game and therefore does not involve the complications associated with the option to cooperate. The players were modeled using very simple two layer neural networks rewarded by adding 1 and punished by subtracting 1 from the connection weights (all of which started with a

weight of 0). The inputs to the network were the opponent's previous moves (referred to as lags), and the outputs were the moves the player would make on the current play. The goal in creating these networks was to use the simplest possible model of sequential dependency detection.

The simulations revealed that processing more lags is an advantage. That is, a network that processed the last two lags (a lag 2 network) would reliably win against a network that processed only the last lag (a lag 1 network). Also a network that treated ties as losses (an aggressive network) could reliably win against a network that was neither punished nor rewarded for ties (a passive network). Furthermore, these effects were additive and approximately equal in magnitude. Another important finding was that the interaction between the networks produced a chaos-like behavior that made them appear to be playing randomly. Subsequent to examining the play of the neural network models, West and Lebiere (2001) investigated the play of humans against the models. They found that humans could reliably beat both the aggressive lag 1 network and the passive lag 2 network. This suggested that humans play similarly to the aggressive lag 2 network. Although there was a small but statistically significant tendency for people to lose against the aggressive lag 2 model rather than tie, this was attributed to the humans being unable to play as consistently as the network model. This interpretation was supported by the fact that subjects reported getting frustrated when playing the aggressive lag 2 network (i.e., playing hundreds of trials is only fun if you are winning). Both the network model and the game theory solution predicted that, on average, people would play each of the three play options with equal frequency. However, the game theory

model (i.e., the combination of the game theory player model and the game theory solution) predicted that people would tie against the networks, which was not the case.

### *Aggregate Behavior*

The West and Lebiere (2001) results show that the sequential dependency player model can account for results in simple zero-sum games that game theory model cannot. In addition, the sequential dependency player model is consistent with the empirical facts. Specifically, people are poor at being random and poor at learning optimal move probabilities because they are instead trying to detect and exploit sequential dependencies. However, the game theory solution did correctly predict that, on average, humans played paper, rock, and scissors with approximately equal frequency. This raises the question of whether game theory can still be considered viable for predicting aggregate move probabilities for this type of game. That is, regardless of the details of how people play, does game theory capture certain higher-level stochastic properties of game playing behavior? After all, even if people do not process game information in the manner suggested by the game theory player model, it may still be the case that across time and across individuals, human game playing can legitimately be viewed as (pseudo) randomly emitting moves according to certain probabilities. To test this possibility and to probe deeper into the relationship between game theory and the sequential dependency player model we tested a variant of PRS.

Rock=2

An aspect of PRS that makes it a very simple game is that each of the three play options is functionally identical to the other two. That is, each move beats one of the

other two moves and loses to the remaining move. In addition, a win is worth the same amount for each move. Thus, it is not surprising that the game theory solution for playing PRS is to play the three options with equal probabilities. The reason this is somewhat problematic is that the agreement between the game theory solution and human behavior for this game may be an artifact of the simplicity and symmetry of the game. To clarify this issue, a modified version of PRS was developed. The new game was identical to the original PRS game except that a win using rock counted for 2 points while a win with scissors or paper counted for only 1 point. In this way, each of the three choices were unique. Rock could win two points and lose only one, scissors could lose two points and win only one, and paper could win or lose only one point. The game theory solution to this modified version of Paper, Rock, Scissors differs depending on whether a zero-sum or non-zero-sum interpretation of the game is adopted. That is, whether a player is trying to maximize the difference in points between himself and the other player, or whether each player is attempting only to maximize the total number of points for themselves. However, since we instructed our subjects to try to maximize the points difference we will focus on the zero-sum interpretation. In this case, the game theory solution is to play paper 50% of the time, rock 25% of the time, and scissors 25% of the time, for the expected outcome of a tie.

### *The Simulated Opponents*

For this study we used the same simple network models as West & Lebiere (2001) and created three simulated opponents for our human subjects to play against. The first two opponents were taken directly from the West & Lebiere (2001) study. They were,

the aggressive lag 2 model and the aggressive lag 1 model. We did this to test the hypothesis that people simply try to maximize wins in this type of game. If this were the case then the results against these two models would replicate the results of West & Lebiere (2001) as neither the humans nor the models would be influenced by rock wins being worth more points. To create the third simulated opponent we adapted the aggressive lag 1 model so that it rewarded the relevant connection weights by 2 instead of 1 when it won with rock. This model was created to pit the human players against a model that might better take advantage of winning with rock, but still had some weaknesses that humans could exploit (i.e., it was only a lag 1 network and it was not set to avoid losing with scissors thereby allowing the opponent to win with rock). The reason for this was that games where humans win are much more informative than games where humans lose, as the loss can be attributed to extraneous factors such as lack of effort, boredom, or frustration. To distinguish this model we will refer to it as the rock=2 lag 1 model.

### *Method*

Ten human subjects played against each of the three network models. They were instructed to try to maximize the points difference in their favor by as much as possible. This goal corresponded to the zero-sum interpretation of the game. The subjects were also told that the network models did not play randomly and that they could be beaten. Additionally, the subjects were instructed to play naturally, not to play too slowly, nor to think too much about their play. The order in which the subjects played the network models was random. Subjects were required to play one game (300 trials) against each of

the three different network models. This process took from between 30 to 45 minutes in total depending on the speed at which the subjects played.

### *Human Results and Discussion*

The points differences between each of the human players and each of the network models were calculated for each trial by subtracting the network score from the human score. Thus a positive score indicated that the human was ahead and a negative score indicated that the network was ahead. The mean total points difference at the end of each game (see Table 25) revealed that the humans were able to win against all of the network models. To test the significance of this we ran a regression on the group points difference data for each different type of opponent across trials. The regression coefficients thus corresponded to the average rate of points accumulation (i.e., points difference/trials) for the humans against each network opponent (the intercept was forced through zero). 95% confidence intervals for the coefficient values revealed that all of them were significantly above zero. That is, against each network models, there was a significant tendency for the humans to win (see table 26).

The fact that people could beat the aggressive lag 2 model under these conditions, whereas they tended to lose in West and Lebiere (2001), where all three varieties of wins were of equal value, indicates that they were able to exploit the fact they knew that wins using rock were rewarded with 2 points. Thus, the hypothesis that people simply try to maximize the number of wins regardless of the number of points awarded for wins, was refuted. That is, the profiles of the humans' play suggested wins with rock were preferred to wins with paper or scissors.

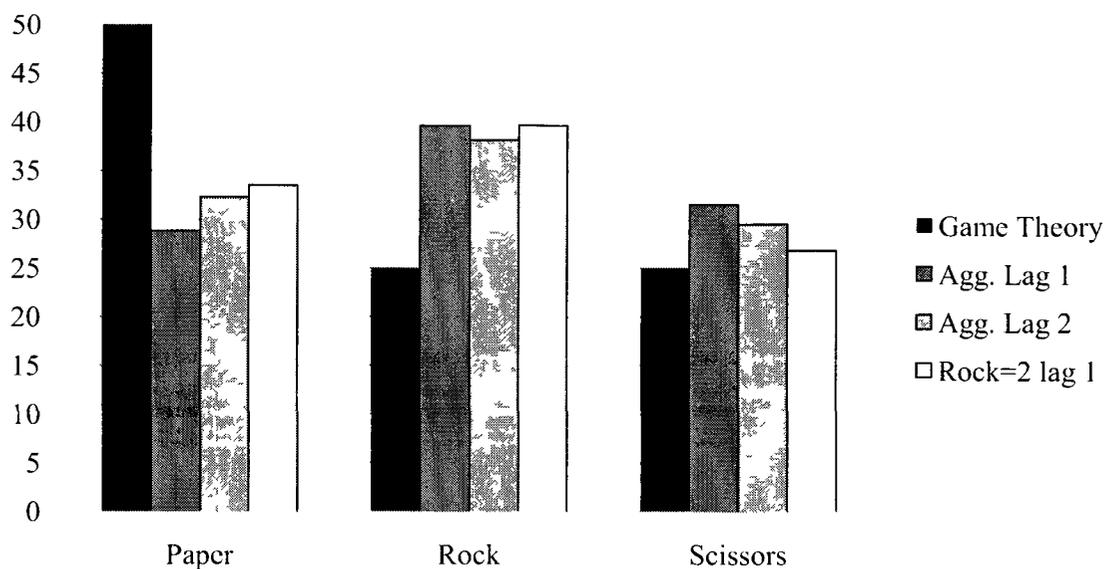
Given that people were sensitive to the payoff information the next question was whether, as in West and Lebiere (2001), the game theory solution predicted the move probabilities for the human players. Figure 58 displays the probabilities for playing paper, rock and scissors for the human subjects, for each of the opponents they faced, with 95% confidence intervals. Figure 58 also displays the predicted probabilities from the game theory solution. As can be seen, the game theory solution was significantly different from the human probabilities.

*Table 25.* The game results of the human versus the neural network models

Network Model	Mean points	Expected points	Strategy Index
	difference	difference	
Lag 1	16.5	4.86	11.63
Lag 2	5.7	11.38	-5.68
Rock=2 lag 1	25.6	5.76	19.84

*Table 26.* Regression analysis on the performance of human subjects against the three network models

Network Model	Regression		Confidence Intervals	
	Coefficient	R Squared	Lower 95%	Upper 95%
Lag 1	0.0886	0.0208	0.0835	0.0938
Lag 2	0.0212	-0.0163	0.0160	0.0264
Rock=2 lag 1	0.0539	0.0251	0.0488	0.0590



*Figure 58.* A comparison between the game theory solution and observed human play probabilities.

We also examined whether the human results could be explained by using the game theory player model without the optimal game theory solution. That is, did the human subjects win by using move probabilities that exploited non-optimal move probabilities produced by the network models? Table 25 shows the average difference in score along with what the difference in score would be if it were determined solely by the overall move probabilities of the two players. The strategy index number is the difference between these two. A score of zero on the strategy index would indicate that the score difference could be accounted for entirely by move probabilities. In all cases the strategy index was significantly different from zero ( $p < 0.05$ , determined by confidence intervals). Given the move probabilities, against the two lag 1 models the humans played significantly better than expected, while against the lag 2 model they

played significantly worse than expected. This can be interpreted as the humans being better at exploiting sequential dependencies than the lag 1 models, but not as good as the lag 2 model. This agrees with the results of West and Lebiere (2001) and suggests that humans were able to narrowly beat the lag 2 because of their knowledge of the payoffs.

### Modeling the Human results

Our next step was to construct a neural network model of how the humans played. For the model we assumed that people detect sequential dependencies in a way similar to a lag2 network. Although the results of this paper and West and Lebiere (2001) show that in games against the lag 2 network, humans seem to be slightly worse at detecting sequential dependencies, we again assumed this was due to humans finding the lag 2 less fun to play against because it is a stronger opponent. In both studies the advantage for the lag 2 network was relatively small. Additionally, West and Lebiere (2001) found that they could account quite well for the results of the other games by modeling humans as aggressive lag 2 networks. To account for the findings in this study we modeled people as aggressive lag 2 networks with the ability to adjust their rewards and punishments so as to best take advantage of the payoffs in the game.

To get an idea of how people could be adjusting the rewards and punishments we obtained self-reports on the strategies used by several of our more successful subjects. These reports generalized to favoring rock wins to paper wins, and paper wins to scissors wins. That is, they were focused first on getting rock wins and second on blocking the opponent from getting rock wins. With this in mind we ran a genetic algorithm to find a

system of rewards and punishments for the neural network model that would match the human point difference results. The result was the following: rock wins = 3, paper wins = 2, scissors wins = 0; rock tie = -1, paper tie = -1, scissors tie = 0; and -3 for all losses. Note that not rewarding scissors wins makes sense as winning frequently with scissors would be associated with the opponent playing paper more often and that would block rock.

The model was played against each of the three networks that the humans faced. Each simulation consisted of 1000 games of 300 trials each. For each game, the net points difference between the models, the probabilities by which the human model selected each of the three play choices, and the strategy index for the human model was recorded. These three measures were used to determine how well the model fit the human data.

### *Model Results*

The results showed that the model matched the point differences well (see Figure 59). The model also reproduced the human move probabilities against each opponent with a high degree of accuracy. The correlation between the model move probabilities and the human move probabilities was 0.964 ( $p < 0.0005$ ). Also, the model provided a good overall fit to the human strategy index data (see Figure 60). Although the model did not match the data as well when the opponent was the lag 2 network, this is actually consistent with our position that humans do not detect sequential dependencies as well against the lag 2 network due to confounding factors, which would not apply to the network model of human behavior. Also, the fact that the model was able to match two

sets of results that it was not explicitly designed to match (the move probabilities and the strategy index values) suggests that the results were reasonably robust.

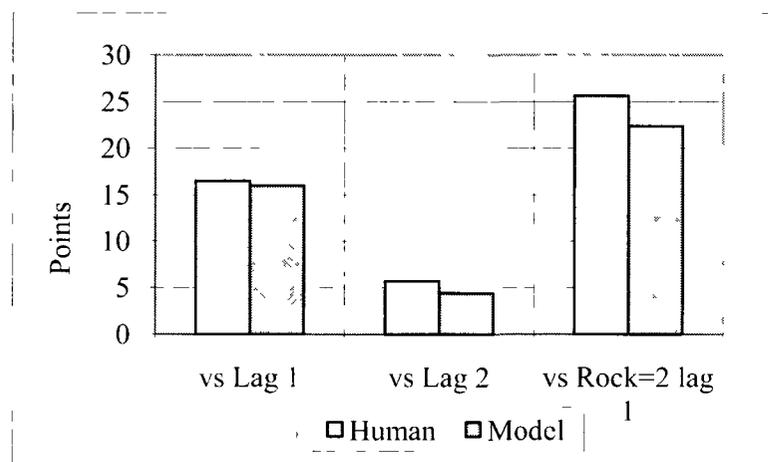


Figure 59. Points differences versus the three network opponents for both human subjects and the model.

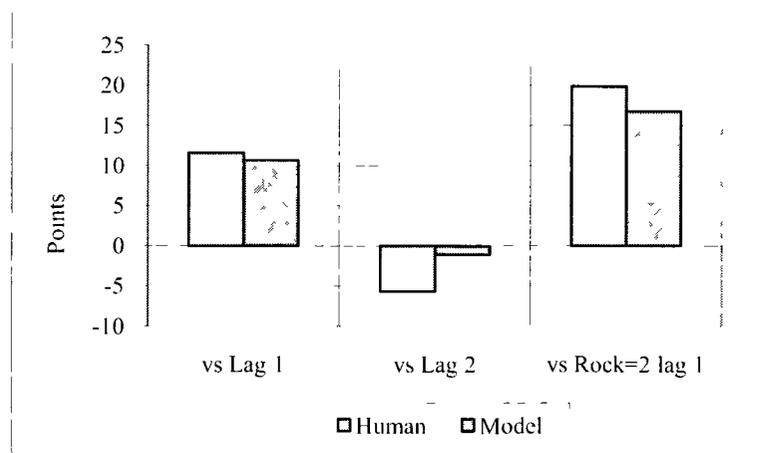


Figure 60. Strategy indices versus the three network opponents for both human subjects and the model.

## Conclusions

The results of this study replicate the West and Lebiere (2001) findings that the commonly used probabilistic game theory model (as defined in this paper) cannot account for the game results when humans play against agents programmed to play by exploiting sequential dependencies. We also demonstrated that when the game payoffs are not all equal, the game theory solution does not predict the aggregate move probabilities. We further demonstrated, using the actual move probabilities that the results could not be accounted for by the game theory player model. That is, the actual move probabilities did not predict the final points differences. These results show that the game theory player model, with or without the game theory solution, is fundamentally different from the way people process information in this type of situation.

In terms of modeling, we replicated the West and Lebiere (2001) result that this type of human game playing can be accurately modeled using simple lag 2 networks. Furthermore, we extended the original model by showing that people are sensitive to different game payoffs and that this can be modeled by adjusting the rewards and punishments associated with different play outcomes. These results are also consistent with a number of ACT-R studies showing that people play a variety of games using the lag 2 strategy (PRS: Lebiere & West, 1999; non-zero-sum games: Lebiere, Wallach, & West, 2000; baseball: Lebiere, Gray, Salvucci, & West, 2003). The ACT-R model works by using the ACT-R declarative memory system as a neural network for detecting sequential dependencies, and produces results similar to the simple networks we used (Lebiere & West, 1999). An ACT-R model equivalent to the one in this paper could be created by “popping” the “chunks” representing sequential dependency patterns a

different number of times for different outcomes. Likewise, a genetic algorithm could be used to fit the model. However, this approach would sidestep the next important issue, which is modeling how humans adjust their reward structure in response to the game payoffs.

#### References

- Estes, W. K. (1972). Research and theory on the learning of probabilities. *Journal of the American Statistical Association*, 67, 81-102.
- Fudenburg, D., & Levine, D. K. (1998). *The theory of Learning in Games*. Cambridge, MA: The MIT Press.
- Gazzaniga, M. S. (1998). The split brain revisited. *Scientific American*, July, 50-55.
- Lebiere, C., Gray, R., Salvucci, D., & West, R. L. (2003). Choice and learning under uncertainty: A case study in baseball batting. *Proceedings of the 25<sup>th</sup> Annual Meeting of the Cognitive Science Society*, 704-709.
- Lebiere, C., Wallach, D., & West, R. L. (2000). A memory-based account of the Prisoner's Dilemma and other 2x2 games. *Proceedings of the Third International Conference on Cognitive Modeling*.
- Lebiere, C., & West, R. L. (1999). A dynamic ACT-R model of simple games. *Proceedings of the 21<sup>st</sup> Annual Conference of the Cognitive Science Society*, 296-301. Mahwah, NJ: Erlbaum.
- Pool, R. (1995). Putting game theory to the test. *Science*, 267, 1591-1593.
- Poundstone, W. (1992). *Prisoner's Dilemma*. New York: Doubleday.

- Restle, F. (1966). Run structure and probability learning: Disproof of Restle's model. *Journal of Experimental Psychology*, 72, 382-389.
- Ritter, F. E., & Wallach, D. P. (1998). Models of two-person games in ACT-R and Soar. *Proceedings of the Second European Conference on Cognitive Modelling*. 202-203. Thrumpton, UK: Nottingham University Press.
- Rose, R. M., & Vitz, P. C. (1966). The role of runs of events in probability learning. *Journal of Experimental Psychology*, 72, 751-760.
- Samuelson, L. (1997). *Evolutionary games and equilibrium selection*. Cambridge, MA: MIT Press.
- Vitz, P. C., & Todd, T. C. (1967). A model of learning for simple repeating binary patterns. *Journal of Experimental Psychology*, 75, 108-117.
- VonNeumann, J., & Morgenstern, O. (1944) *Theory of games and economic behaviour*. Princeton, NJ: Princeton University Press.
- Wagenaar, W. A. (1972). Generation of random sequences by human subjects: A critical survey of the literature. *Psychological Bulletin*, 77, 65-72.
- Ward, L. M. (1973). Use of markov-encoded sequential information in numerical signal detection. *Perception and Psychophysics*, 14, 337-342.
- Ward, L. M., Livingston, J. W., & Li, J. (1988). On probabilistic categorization: The Markovian observer. *Perception and Psychophysics*, 43, 125-136.
- West, R. L. (1998). Zero sum games as distributed cognitive systems. *Proceedings of the Complex Games Workshop*. Tsukuba, Japan: Electrotechnical Laboratory Machine Inference Group.

West, R. L., & Lebiere, C. (2001). Simple games as dynamic, coupled systems: Randomness and other emergent properties. *Cognitive Systems Research*, 1(4), 221-239.

## ACT-R VERSUS NEURAL NETWORKS IN ROCK=2 PAPER, ROCK, SCISSORS

Previously published as Rutledge-Taylor, M. F. & West, R. L. (2005). ACT-R versus neural networks in rock=2 paper, rock, scissors. *Proceedings of the Twelfth Annual ACT-R Workshop*, 19-23. Trieste, Italy: Universita degli Studi di Trieste.

### Contribution

This paper presents an ACT-R model of the Rock=2 Paper, Rock, Scissors experiments. These results are referenced in Using DSHM to Model Paper, Rock, Scissors.

### Abstract

Recent research on cognitive modeling and game playing has focused on the game of Paper, Rock, Scissors (PRS). Models of PRS players have been created using both neural networks (West & Lebiere, 2001) and ACT-R (Lebiere & West, 1999; West, 1998). In all cases successful models of human play were created. This seems to be, in part, because of the simplicity of the game. In Rutledge-Taylor & West (2004) neural network models of players of a modified version of paper, rock, scissors were tested. The network model was able to fit the human data, but it was necessary to use a genetic algorithm to adjust the reward and punishment amount for the various game outcomes. The ACT-R model is much more constrained than a generic neural network in terms of how it can be adjusted. Since it uses the declarative memory system, learning is based on "harvesting" and "popping." In the present work the question of how to create an ACT-R model where rewards of various magnitudes need to be implemented is investigated.

This is done by exploring some simple techniques such as the double retrieval and harvesting of chunks, and the manipulation of the default parameter for noise (ans). The results are compared to the human data from Rutledge-Taylor & West (2004) in which humans played the variant of PRS described above.

#### Introduction: The Neural Network Models

Neural network models of human paper, rock, scissors game play have been described in Rutledge-Taylor & West (2004), and West & Lebiere (2001). The game of paper, rock, scissors was chosen for two reasons. It is a game familiar to most prospective experimental participants. Due to its simplicity, an analysis of how it is played is tractable. In Rutledge-Taylor & West (2004), and West & Lebiere (2001) the same types of neural networks were used. The networks were perceptron-like in that they had no hidden layer. The output layer consisted in three nodes, one for each of the possible play options of paper, rock, and scissors. When presented with input, the play option associated with the output node with the greatest activation is chosen by the network. The input layer consisted in either one or two groups of three binary nodes, one for each of the possible play options. Each input group represented a move made by the model's opponent in the past history of a game underway. Models with three input nodes were called lag 1 models; they received as input the last move made by their opponent. Models with six input nodes were called lag 2 models; they received as input the last two moves made by their opponent. For each input group, the node corresponding to the move made would be have an activation of one, while the other two would have activations of zero. For example, if a lag 2 model's opponent had played paper last and

scissors the time before that, the input pattern to the network would be  $((1,0,0),(0,0,1))$ . The network weights were integer values, which in the case of West & Lebiere (2001) were initialized to 0, and in the case of Rutledge-Taylor & West (2004) were randomly initialized to  $-1$ , 0, or 1. For each model there is an associated three by three reward matrix which determines how the model's network weights should be adjusted after an iteration of play based on the outcome. Each cell in the matrix corresponds to the combination of an outcome from the model's perspective {win, tie, loss} and the move chosen by the model {paper, rock, scissors}. The value in the corresponding cell is added to the network weights that contributed to making the selection of the move played. The two most simple reward matrices were called aggressive and passive. In the passive reward matrix, cells associated with wins have values of 1, ties have values of 0, and losses have values of  $-1$ . The aggressive matrix is identical, with the exception that ties have values of  $-1$ . Thus, networks with passive reward matrices, hereafter, referred to simply as passive networks or passive models, treat ties as neutral events whereas the aggressive networks categorizes both ties and losses as non-win negative events.

West & Lebiere (2001) pitted human participants against several of the neural network models, and compared these results to model versus model games. They found that pairs of identical models, when pitted against one another, on average, tied; networks that processed two lags had a competitive advantage over networks that processed only one; and, aggressive networks had an advantage over passive networks. Interestingly, the extra lag and aggressive advantages were approximately equal in magnitude. Humans participants showed a performance profile similar to that of the aggressive lag 2 network model. On average, the human participants had won 9.99 more games than the

aggressive lag 1 models after 300 games, and against the passive lag 2 models had won 11.14 more games after 287 games. However, against the aggressive lag 2 model, the network had, on average, a 8.89 win advantage after 20 minutes of play. The failure of the human participants to, at least, tie the aggressive lag 2 networks was attributed to a lack of motivation on the part of the human players; playing an opponent that is difficult to beat is less fun than playing one that can be taken advantage of.

Rutledge-Taylor & West (2004) investigated the performances of humans against computer neural network models in a slightly different version of paper, rock, scissors. In this version, called Rock=2, two points are awarded to the winner in the rock versus scissors combination of play, and one point is awarded to the winner in the paper versus rock, and scissors versus paper cases. Humans played against the aggressive lag 1 model, the aggressive lag 2 model, and new model called the rock=2 model. The rock=2 model was identical to the aggressive lag 1 model, with the exception that the rock=2 reward matrix cell corresponding to winning with rock had a value of 2. The standard aggressive models are not explicitly sensitive to the game condition that winning with rock is worth more than winning with either paper or scissors. The rock=2 model was designed to be, potentially, sensitive to this new aspect of the game.

The results of play were that, after 300 games, the human participants were, on average, able to earn 16.5 more points than the aggressive lag 1 models, 5.7 more points than the aggressive lag 2 models, and, 25.6 more points than the rock=2 model. Additionally, the expected points difference based on the ratios with which the players played each of the three possible moves was calculated. This expected points difference was subtracted from the actual points difference to produce a measure of the human

players' abilities to orchestrate more wins than would be the case if the humans played randomly according to the ratio by which they made their choices of play. This measure was called the strategy index. Against the aggressive lag 1, the humans scored a strategy index of 11.6; against the aggressive lag 2, -5.7; and, against the rock=2 model, 19.8. This suggests that against the lag 1 models, the humans were able to predict the models' moves with greater than chance success. However, against the lag 2 model, it was the neural network that was better able to predict the humans' moves. That the humans enjoyed an average net points advantage was entirely due to their sensitivity to the fact that winning with rock was worth two points (i.e., they would play so as to maximize wins using rock).

In Rutledge-Taylor & West (2004) a neural network model of human play was proposed. It was produced using a genetic algorithm. The model was a lag 2 network with a unique reward matrix: rock wins = 3, paper wins = 2 scissors wins = 0; rock tie = -1, paper tie = -1, scissors tie = 0; and -2 for all losses. This model produced game results similar to that of the human participants; see table 27.

*Table 27. Human and GA neural network model performances compared*

Opponent	Human		GA model		Disagreement	
	Pts. Diff	S.I.	Pts. Diff	S.I.	Pts. Diff	S.I.
Agg. Lag 1	16.5	11.6	16.98	10.68	-0.48	0.92
Agg. Lag 2	5.7	-5.7	4.42	-1.09	1.28	4.61
Rock=2	25.6	19.8	22.37	16.78	3.23	3.02

This GA model produced good results. However, there is, at best, a rather ad hoc story to be told about the values in the model's reward matrix. This concern motivated the production of an ACT-R model of human paper, rock, scissors play.

### ACT-R Models

There are many different ways to build a paper, rock, scissors game player in ACT-R. Not only are there many parameters, affecting the manner in which ACT-R models behave, there are also several different architecturally distinct ways that a model designed to play paper, rock, scissors could be built. For example, there is the distinction between a rule based ACT-R model and an exemplar based ACT-R model, as described in Anderson & Matessa (1998). To examine every architecturally distinct ACT-R model would be too grand a project, so we limited ourselves to exemplar based models of game play proposed by Lebiere and West (1999). We examined both lag 1 and lag 2 ACT-R models of the normal rock=1 version of paper, rock, scissors, and the rock=2 version of the game.

The model is embedded in LISP code. The LISP code provides the ability to automatically reset and run the model multiple times, and to log data from those runs. It is also the means of sustaining an opponent for the ACT-R model. A LISP implemented neural network model is defined in the code.

#### *How does the Model Work?*

As mentioned above the ACT-R models are exemplar based. This means that the models make their moves based on predictions of what sequences of opponents' moves they've experienced in the past. There is a chunk in declarative memory corresponding to each

distinct pattern of moves plus a prediction. For example, in the case of the lag 2 model, there are 27 such chunks; there are nine combinations of last and next to last move, and for each combination there are three possible predictions. For each iteration of play, the model recalls a chunk that matches the last two moves made by the LISP neural network opponent; the chunk with the greatest sum of activation and noise, is selected. The move that beats the predicted move is played, and the LISP network makes its choice (as described above). The result of the play is recorded (using LISP). Were this all that the model did, it would not learn. This is because only the recalled prediction would be rewarded, and worse, regardless of whether the prediction was correct. Therefore, after the network's move is revealed, a series of productions retrieve the chunk in memory corresponding to this move (i.e., the chunk triplet of last move, second to last move, and the networks subsequent move). This way, the correct chunk is reinforced.

#### *Testing the ACT-R Models*

In Lebiere & West (2001), humans were pitted against three different neural network models, as described above. In the case of the human versus the aggressive lag 2 model, the results were somewhat ambiguous. Whether the score difference in favour of the model was due to an inherent skill inferiority on the part of the human participants, or due to extraneous factors such as lack of motivation is unclear. Therefore, the ACT-R models of human play in the standard paper, rock, scissors game, were compared only to the results of humans versus the aggressive lag 1 model and the passive lag 2 model. For both the lag one and lag two ACT-R models, ans values of 0.35, 0.30, 0.28, 0.25, and 0.15, were tested. Also, the effect of optimized learning was tested. For each

combination of parameters, 100 simulations of 300 games each were run. The models were reset between each simulation. The first obvious result was that optimized learning drastically reduced the performance of the ACT-R model. No combination of lag and noise value could produce an ACT-R model that came close to replicating the human data. However, when optimized learning was turned off (sgp :op NIL), several good models of human play were produced. See table 28 for a comparison of two models.

*Table 28.* Naive ACT-R models of rock=1 paper, rock, scissors play

Opponent	Human	ANS=0.28, OL=NIL		ANS=0.35, OL=T	
	Win. Diff	Win. Diff	Error	Win. Diff	Error
Agg. Lag 1	9.990	12.297	2.307	4.337	-5.653
Pas. Lag 2	11.645	8.149	-3.496	-3.010	-14.655
Sum SQ. Err.			17.548		246.727

Optimized learning drastically compromises the ACT-R models' abilities to compete against lag 2 neural network models (these models also performed poorly against the aggressive lag 2 model in pilot simulations). The best ACT-R model was the lag 2 model with a noise setting of 0.28, and optimized learning turned off. Although, the win differences against the two opponent neural networks was not a perfect match, this model is considered a success due to the fact that only a single parameter was manipulated; all of the other parameters were left at their default settings (with the exception of the optimized learning parameter, which is by default on).

Coming up with an ACT-R model of the Rock=2 paper, rock, scissors player proved to be a challenge. As a starting point, the model with a noise setting of 0.28 and optimized learning turned off was pitted against the aggressive lag 1, the aggressive lag 2, and the Rock=2 models. The result was a fairly good match of the human data. Next, a model with the default noise setting (sgp :ans 0.25) was tested. This model was a somewhat better match.

Table 29 presents a comparison of the points differences and strategy indices for human participants, and the two ACT-R models designed to play the Rock=1 game. The Sum SQ diff. row indicates the sum of the squares of the differences between the models' scores and the human data. The rating row is the sum of the points difference and double the strategy index difference, divided by 1000. The strategy indices were weighted more heavily in determining the rating of a model due to the fact that its values tended to be smaller than the points difference values.

*Table 29.* Naive ACT-R models of rock=2 paper, rock, scissors play

Opponent	Human		ANS=0.28, OL=NIL		ANS=0.25, OL=NIL	
	Pts. Diff.	S.I.	Pts. Diff.	S.I.	Pts. Diff.	S.I.
Agg. Lag 1	16.5	11.6	16.921	16.781	11.554	14.748
Agg. Lag 2	5.7	-5.7	-7.762	-5.198	-4.267	1.877
Rock=2	25.6	19.8	41.396	32.153	35.792	27.309
Sum SQ diff.			430.928	179.689	227.684	80.901
Rating			0.079		0.039	

Despite the fairly good match of the Rock=2 models to the human data, there was one main concern. This was the fact that the naïve models lost to the aggressive lag 2 model. This, however, should not be unexpected. In Lebiere & West (2001) it was observed that human participants tended to lose to the aggressive lag 2 model. However, in Rutledge-Taylor & West (2004) it was reported that human participants were aware that winning with rock was more valuable than winning with paper or scissors, and explicitly tried to maximize rock wins. This is apparent from the fact that human participants were able to achieve, on average, a positive points difference against the aggressive lag 2 model, despite the negative strategy index. Thus, an ACT-R model sensitive to the Rock=2 game parameters was designed.

Just as there are many different ways to build a paper, rock, scissors player in ACT-R, there are various options for how to “build-in” knowledge that winning with rock is worth more than winning with either paper, or scissors. The option that was chosen for this experiment was to double harvest chunks associated with particular plays by the model's opponent. Three variations were tested. First, rock plays received extra attention. That is, when the neural network played something other than rock, the relevant chunk was retrieved and harvested once, just as is the case with the Rock=1 models. In the case of rock, this retrieval process occurs twice. The rationale is that human participants might be merely paying more attention to when their opponents play rock. Technically, the effect is that the ACT-R model will be more likely to predict that the neural network opponent will play rock, and respond with paper. This will cause the ACT-R model to play paper more frequently, which will indirectly have the effect that the neural network will be more likely to play scissors, which is desirable for the ACT-R

model. This is because by playing scissors more frequently, there is a greater likelihood that ACT-R will benefit by playing rock. Obviously, the dynamics of the interplay between the players is very complex. In fact, Lebiere & West (2001) argue that when two neural network players are pitted against one another, they form a chaotic dynamic system. Second, we tested models which gave extra attention to scissors. And last, we tried a model that gave extra attention to both rock and scissors plays by its opponent.

In testing the ACT-R models in the Rock=2 game, optimized learning was turned off, and a noise setting of 0.25 was used.

### The Results

Of the three ACT-R models, each paying extra attention to different subset of opponent moves, none matched the human data perfectly. However, the model paying extra attention to opponent plays of both rock and scissors produced a good qualitative fit to the human data. Figure 61 depicts a comparison of the points difference achieved by both the human participants and the Rock and Scissors ACT-R model versus each of the three neural network models. Table 30 summarizes the comparisons of the models' performances against the three neural network opponents and that of human experimental participants. Although the "scissors" model matched the strategy index values best, its failure to beat the aggressive lag 2 neural network is the principle reason for disqualifying it as an adequate model of human play.

Table 30. Savvy ACT-R models of rock=2 paper, rock, scissors play

Opponent	Rock		Scissors		Rock+Scissors	
	Pts. Diff.	S.I.	Pts. Diff.	S.I.	Pts. Diff.	S.I.
Agg. Lag 1	37.7	40.64	4.09	19.14	26.24	21.63
Agg. Lag 2	11.88	15.36	-16.4	-1.38	7.52	2.23
Rock=2	52.94	56.94	25.38	30.02	38.45	29.07
Sum SQ diff.	1235.22	2666.24	642.45	180.08	263.35	249.54
Rating	6.57		1.00		0.76	

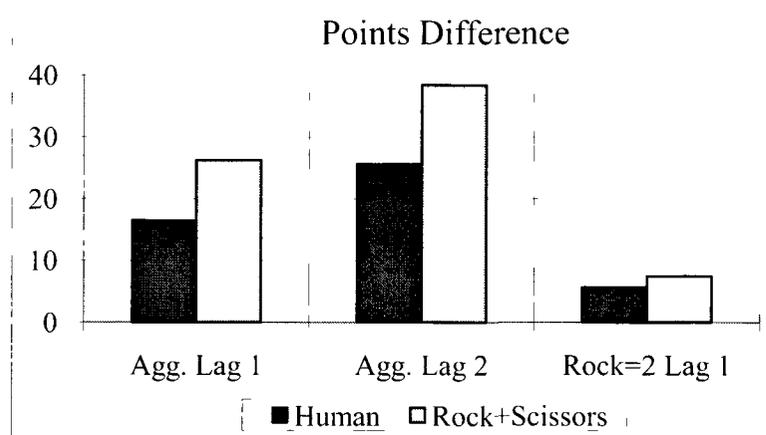


Figure 61. Human and model points differences versus the three network opponents.

### Conclusions

The results of human experimentation from Lebiere & West (2001) and Rutledge-Taylor & West (2004) were replicated with varying degrees of success. Good models of the human performance in the Rock=1 game were produced, without tinkering with the available parameters controlling the manner in which ACT-R behaves. However, in the

case of Rock=2 game, the naïve ACT-R model was unable to take advantage of the fact that winning with rock was worth more than winning with either paper or scissors. Of three models designed to pay extra attention to certain opponent moves, the model that attended to both rock and scissors plays matched the human data well. Therefore, the process of retrieving chunks twice is a viable option for increasing the activation of chunks more than would be achieved by the normal “popping” and “harvesting” process. Intuitively there seems to be a need for this.

#### References

- Anderson, J. R. & Matessa, M. (1998). The rational analysis of categorization and the ACT-R architecture. In M. Oaksford & N. Chater (Eds.), *Rational models of cognition*. Oxford University Press.
- Lebiere, C., & West, R. L. (1999). A dynamic ACT-R model of simple games. *Proceedings of the 21<sup>st</sup> Annual Conference of the Cognitive Science Society*, 296-301. Mahwah, NJ: Erlbaum.
- Rutledge-Taylor, M. F. & West, R. L. (2004). Cognitive modeling versus game theory: Why cognition matters. In M. Lovett, C. Schunn, C. Lebiere & P. Munro (Eds.), *Proceedings of the Sixth International Conference on Cognitive Modeling* (pp. 255-260). Pittsburgh, PA: Carnegie Mellon University/University of Pittsburgh.
- West, R. L. (1998). Zero sum games as distributed cognitive systems. *Proceedings of the Complex Games Workshop*. Tsukuba, Japan: Electrotechnical Laboratory Machine Inference Group.

West, R. L., & Lebiere, C. (2001). Simple games as dynamic, coupled systems: Randomness and other emergent properties. *Cognitive Systems Research, 1(4)*, 221-239.