A COMPUTATIONAL MODEL OF MEMORY PROCESSES IN THE EXPECTATION-VIOLATION EFFECT

NILÜFER ÖZYÖRÜK

DECEMBER 2004

A COMPUTATIONAL MODEL OF MEMORY PROCESSES IN THE EXPECTATION-VIOLATION EFFECT

A THESIS SUBMITTED TO

THE GRADUATE SCHOOL OF INFORMATICS

OF

THE MIDDLE EAST TECHNICAL UNIVERSITY

BY

NILÜFER ÖZYÖRÜK

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE

OF

MASTER OF SCIENCE

IN

THE DEPARTMENT OF COGNITIVE SCIENCE

DECEMBER 2004

Approval of the Graduate School of Informatics

Assoc. Prof. Dr. Nazife Baykal
Director

I certify that this thesis satisfies all the requirements as a thesis for the degree of Master of Science.

Prof. Dr. Deniz Zeyrek
Head of Department

This is to certify that we have read this thesis and that in our opinion it is fully adequate, in scope and quality, as a thesis for the degree of Master of Science.

Assist. Prof. Dr. Bilge Say
Supervisor

Prof. Dr. Hasan Gürkan Tekman
Co-Supervisor

Examining Committee Members

Assoc. Prof. Dr. Cem Bozsahin

Assist. Prof. Dr. Bilge Say

Prof. Dr. Hasan Gürkan Tekman

Dr. Meltem Turhan Yöndem

Prof. Dr. Deniz Zeyrek

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Nilüfer Özyörük

ABSTRACT

A COMPUTATIONAL MODEL OF MEMORY PROCESSES IN
THE EXPECTATION-VIOLATION EFFECT

Özyörük, Nilüfer

MS, Department of Cognitive Science

Supervisor: Assist. Prof. Dr. Bilge Say

Co-Supervisor: Prof. Dr. Gürkan Tekman

December 2004, 141of pages

This thesis focuses on modeling Expectation-Violation Effect, which is the superior recall of weakly associated pairs of words over strongly associated pairs. The goal of this thesis is to provide an exploratory computational model. A virtual experiment is conducted based on the datasets used in the psychological experiment by Amster et al. (1992). The computational modeling of this phenomenon is carried in the medium of ACT-R cognitive architecture.

Keywords: Expectation-Violation Effect, Isolation Effects, Cognitive Architecture, ACT-R, Cognitive Modeling, Association

ÖZ

BEKLENTI-KIRIKLIGI ETKISININ ARKASINDAKI ZIHINSEL SÜREÇLERIN BILISIMSEL MODELLEMESI

Özyörük, Nilüfer

Master, Bilissel Bilimler Bölümü

Tez Yöneticisi: Assist. Prof. Dr. Bilge Say

Ortak Tez Yöneticisi: Prof. Dr. Gürkan Tekman

Aralik 2004, 141 sayfa

Bu tez, zayif çagrisimi olan kelime çiftlerinin güçlü çagrisimi olan çiftlerden daha iyi hatirlanmasi demek olan Beklenti-Kirikligi Etkilerinin modellenmesi üzerinedir. Bu tezin amaci, baslangiç niteliginde bir ön bilgisayar modeli üretmektir. Çalisma, Amster et al. (1992) psikolojik deneyinde kullanilan veri setleriyle bu süreçlerin bilissel modelini gerçeklestirmistir. Bu sürecin bilissel modellemesi ACT-R bilissel mimarisinde gerçeklestirilmistir.

Anahtar Kelimeler: Beklenti-Kirikligi Etkisi, Soyutlanma Etkileri, Bilissel Mimari, ACT-R, Bilissel Modelleme, Çagrisim

ACKNOWLEDGEMENTS

I am grateful to my advisor Assist. Prof. Dr. Bilge Say for her guidance. I also would like to extend my thanks to my co-advisor Prof. Dr. Gürkan Tekman. My special thanks go to my parents Mukbil and Sezer Özyörük, and to my friend Evgueni A. Stepanov.

TABLE OF CONTENTS

LIST OF TABLES

TABLE

LIST OF FIGURES

FIGURE

LIST OF ABBREVIATIONS

ABBREVIATION

LAS	Law of Association Strength
EVE	Expectation-Violation Effect
ACT-R	Atomic Components of Thought – Rational
HAM	Human Associative Memory
AEV	Associate-Expectation-Verification

CHAPTER 1

Introduction

Associations are first mentioned by Aristotle and later studied by many like Freud (1856 - 1939) and behaviorists who are inspired by the work of Pavlov (1849 – 1936). Associations are a result of the cognitive ability to connect concurrently occurring elements. An association develops as a result of co-occurrence and the frequency of co-occurrence between two pieces of information. The elements are associated according to some atomistic and mechanical principles. Hume (1817, cited in Leahey 1980) suggested that associations in the mind works just as same as gravity works in the nature.

Over the years, this observation has been investigated intensively from various perspectives. One of the established findings of these studies on associations is the law of association strength. This law suggests that if two items are highly associated, like 'cat-dog', the memory for these items, individually or as a pair, will be much better than if they are weakly associated pairs, like 'cat-stair'. The likelihood of retrieval of one item, when the other item is presented as a cue, will increase with the increasing strength of the association between them. Over the years, the association strength perspective on learning has become an underlying assumption for the theories of memory and cognition.

One of the studies on the association strength and its influence on memory was conducted by Deese (1959) where the subjects were given lists containing 15 words for study and later asked for an immediate free recall. Subjects listened to the words in the lists only once. Eighteen lists were arranged according to their average inter-item associative strength, i.e. whether the words in that list were eliciting other words in the same list. Six of the lists contained pairs which frequently elicited each other as free associates, another six list contained words which had low frequency and the last six contained words which never elicited each other as free associates. After the free recall test, he found not only that association strength was positively related to recall rate in free recall[1], but also that it possibly was a direct and unmediated activity.[2]

Another study, which used children as subjects, was conducted by Palermo and Jenkins (1964a). They tested Jarret and Scheibe's (1963) statement that associative strengths taken from word-association test norms provide an index of preexisting strengths between elements and also determine the rate of learning for paired-associates. They used lists of pairs with high or low associative strength. Subjects were given the lists only once for study and later a cued recall[3] test for retrieval was applied. The results of this study found mean number of errors to vary

[1] Free-recall: A recall test, where the subject is not presented any outside cue and asked to retrieve to-be-remembered items in any order. The difference from cued-recall is that subjects are free to choose their own search strategy to retrieve and need to use their own cues.

[2] Subjects did not seem to utilize mnemonics for better performance in recall.

[3] Cued-recall: A recall test, which presents an external cue to remind the to-be-remembered item.

inversely with the associative strength. They claimed that their findings strongly support Jarret and Scheibe's statements and it can be concluded that association norms can predict rate of learning.

For ease of discussions, this thesis will use the acronym LAS (Law of Association Strength) to refer to increased ease of learnability, or retrievability in a recall test, of the items which are already strongly associated.

1.1 The Problem: Contradiction of Isolation Effects

Isolation effects represent the retrieval superiority of what is called 'isolated items'. A general review of these studies suggests that isolated items are bizarre and/or irrelevant, meaning that they do not fit to our mental schemes or to the background in which they are presented. Isolation effects are a divergence from the predictions by LAS. Instead of the retrieval benefit of the already established memory patterns, as suggested by LAS, isolation effects show that items isolated from the rest in some way, can have retrieval superiority.

An earlier example of isolation effects is von Restorff effect and it is a typical example (Hunt, 1995). An example of what von Restorff effect looks like can be such that, if a list of fruits like: apple, pear, banana, truck, kiwi and grape; are presented to a subject, 'truck' will be better recalled as it will be the surprising item and that it does not fit to the conceptual category of the list. The isolation effects are important as they show the possibility of processes other than the strength of association that could be responsible for learnability of items.

Isolation paradigm consists of a wide collection of phenomena. Natural observation or experimentation, through different methods researchers have found different reasons for the phenomena we group under isolation effects. Yet, there are certain commonalities among the processes behind these isolation effects, which seem to exhibit similar attributes. Generally, they seem to stand-out on a background, are surprising to the perceiver, and seem to use some kind of a cue for retrieval.

Expectation-Violation Effect (EVE[4]) is a member of isolation effects. EVE was observed and studied by Hirshman (1988) over 13 different experiments. In addition to the characteristics of isolation effects as mentioned above, Hirshman also suggested that surprise and its utilization as a retrieval cue, as well as the need to have the isolated items stand out among the common items were critical. Later, Amster, Brooks, Lucas and Özyörük-Gee (1992) have conducted a series of studies that confirmed EVE. Amster and Özyörük within this series of studies have specifically manipulated the strength of associability and its influence on EVE in contrast to classical findings aligned with LAS.

Contrary to the expectations of LAS, EVE shows that stronger association strength does not always predict better retrieval. In EVE weakly associated pairs are better recalled than strongly associated pairs. Nevertheless, EVE does not disprove LAS, but simply implies that when the conditions are changed the processes are also changed, thus they may not be obeying the same laws.

[4] For ease of discussions, an acronym EVE will be used in place of Expectation-Violation Effect, which was not used by Hirshman (1988).

1.2 The Aim and the Scope

The goal of this thesis is to build an exploratory computational model of Amster and colleagues (1992) experiment, using a computational architecture. ACT-R (Atomic Components of Thought -Rational) is the selected architecture for modeling EVE because it is a unified cognitive architecture. ‘Unified’ in the sense that ACT-R does not focus on implementing only certain aspects of learning and cognition, but attempts to integrate all aspects of cognitive phenomena as they become established in the psychology literature.

A model for an explanatory simulation of the memory processes reflecting association strength would benefit from constructing a semantic network. A model without utilizing a semantic network would not be a realistic implementation. A semantic network would provide the model with the ability to precisely observe the predictions about semantic elaboration and such. This is not the route this thesis intends to take; neither does it intend to focus on the intricacies of semantic elaboration or the processes behind the better retrieval of the strongly associated pairs. Building a semantic network for ACT-R, since there is no existing one, would have required an effort well beyond the scope of this thesis.

Without a semantic network, simulation of EVE will have to rely on parameters accounting for association strengths. Thus this model does utilize parameters to account for semantic elaboration or associative processes.[5] Both the weaker pairs, which lead to EVE and the stronger pairs were simulated without a

[5] Behavior of ACT-R models are controlled at the subsymbolic level through settable parameters that refer to the values and variables used in the equations. Besides, there are parameters, also referred to as traces, which are used to control the output of the model.

semantic network. The processes used in the model are not meant to be psychologically realistic in the way they are implemented. Rather, they were partly inspired from the literature on EVE and partly manipulated with ACT-R specific constructs and parameters to obtain a reasonable correlation with Amster and colleagues (1992) results. Further experimentation will be required to clarify such processes. This model however, is the first model of EVE on ACT-R.

The scope of this thesis is limited to the possible cognitive processes leading to EVE, as studied by Amster and colleagues (1992) and to the computational modeling of the possible processes suggested. This study is a mere exploration of an important, but clearly neglected issue: Modeling of Isolation Effects, specifically EVE. An account for the whole of isolation effects is beyond the scope of this thesis, neither it is intended to be a final word in the modeling of EVE. This thesis specifically focuses on and simulates the study by Amster and colleagues on ACT-R architecture.

1.3 The Methodology

Computational modeling is a new phase in the pursuit of knowledge, made available by the emergence of computers. Over the years modeling has become a valuable tool for scientists in exploring the possibilities and the plausibility of psychological theories. Schunn and Wallach (2004) suggest that as the theorizing in science becomes more complex, it increasingly gets to be more important to have mathematical or computational instantiations, in order to determine whether the predictions of verbal theories hold.

Cognitive modeling and thus ACT-R modeling attempts to better understand human intelligence by using computer simulations. But, computers are capable of simulating practically anything. For that reason, it is important to reconsider our theoretical approach. Most of the time, psychologists have been attempting to understand human mind through binary oppositions, i.e. dissociations. Unless, a unifying perspective can be introduced, the continued dissociations can only lead to more questions in science than answers. Modeling allows the consideration of multiple processes and constraints simultaneously. Thus it has the capacity to integrate the narrowed down processes into a picture (Newell 1990, cited in Taatgen, 2005a). Thus computational modeling is a complementary method for scientific study of cognition. It is a useful tool in furthering our scientific pursuits by allowing us safe trial-and-error opportunities to test before we invest in further experimentation.

This thesis will too will attempt to apply this approach. In the experiments on EVE, the stronger and the weaker pairs show opposite effects. So, even though the two pair groups do have to be manipulated somewhat differentially in order to get the different results obtained, the model attempts to draw a picture around the phenomena trying to integrate various factors cited in the literature such as, surprise responses as cue, or blind-alley searches which will all be discussed in Chapter 2.

The aim of this thesis is to implement a preliminary simulation of EVE based on the results of this author's previously conducted study within Amster and colleagues (1992). Hirshman's (1988, 1989) studies are also vital to the theoretical background. So a review on isolation effects, Hirshman's studies and Amster and colleagues' studies will be done before the thesis moves on to explaining the model.

1.4 The Organization of the Thesis

Chapter 2 presents a review of association strength perspective, isolation effects and EVE. The ACT-R architecture and relevant ACT-R models will be discussed in Chapter 3 and the implementation of the model will be presented in Chapter 4. Chapter 5 will contain the conclusion.

CHAPTER 2

Isolation Effects and Related Mental Processes

The following sections overview the issue of establishing a common name for what will be called as 'isolation effects.' Later, a brief overview of the isolation effects will be mentioned. Lastly, EVE as studied by Hirshman (1988) and Amster and colleagues (1992) will be presented.

2.1 A Common Name

There is a long history of research in what can be grouped under 'Isolation effects'. There are many varieties of them, with different names and methods of study. Neither establishing their commonalities nor finding a common name is an easy task.

Isolation effects are known to result from salient and distinctive information, which are usually experienced by the people as bizarre. Therefore, bizarreness, salience, distinctiveness are only few of the popular names. Salience is not the best candidate as Hunt and Lamb (2001) suggest, because salience is more like monitoring the environment for radical changes instead of processing distinctive information. Bizarreness is not a good definition either, because items do not need to be 'bizarre' for the effect to take place. Distinctiveness seems to be the best name, but the definition of distinctive information has been difficult in the literature. Hunt (1995) warns us about the circularity of the definition of distinctiveness. He says that in studies, distinctiveness have normally been defined through the subjects' responses or descriptions of distinctiveness. It is basically a psychological resultant of the processes which lead to discrimination of items. If it was not up to these processes' activities, those items would have been perceived as similar. Thus, distinctiveness itself is not an independent variable. Schmidt (1991) suggests that there are no context-free, subject-free definitions of distinctiveness. He suggests distinctiveness is a hypothetical construct as it depends on the people's own definitions and the context in which it is perceived. Hunt and Lamb suggest that isolation is the most suitable common name for these phenomena as it naturally implies that these better recalled items, which normally would not be retrieved this well, are the result of certain processes leading them to be isolated from the rest of the to-be-remembered material to their benefit in retrieval.

2.2 Types of Isolation Effects: Distinctive Items

Schmidt (1991) reviewed types of phenomena which were discussed as 'isolation effects' based on Hunt and Lamb's (2001) suggestion, in the previous section. But, Schmidt at the time had used the name 'distinctiveness'. So, this section will be labeled with the same name as Schmidt has used.

Schmidt divided distinctiveness related studies into four main categories: Emotional, Primary, Secondary and Processing Distinctiveness.

2.2.1 Emotional Distinctiveness

Emotional distinctiveness effects are triggered by emotionally pronounced situations or upheavals. The relationship between memory and emotion is nothing straightforward. Schmidt (1991) reviewed many studies showing that emotionally arousing information may be poor with immediate recall, but better in later recall. Also, emotional information can be better recalled than neutral information even when the neutral information was studied with an intention of learning. Thus, emotion can, at times, be a much more powerful learning mediator than intention.

The studies investigating emotions and memory, achieve results with trauma, depression, arousal and humor. They all show different effects on memory functions. The most important point about emotional distinctiveness is people's own perceptions of personal relevance or importance of those memories. So it is quite a subjective experience. It is clear that they add a different dimension to memory and according to Easterbrook Hypothesis (Easterbrook, 1959) that is most possibly the contribution of the focus of attention. Attention seems to be an important component of isolation effects, as has been suggested since 1959.

One aspect of emotional distinctiveness seem to be more relevant to EVE: It has been pointed out by Hirshman (1988) that the orienting reflex or responses[6], which are also categorized under emotional distinctiveness, and the attention accompanying this process, seem to carry great similarity with the surprise response that is needed to bring out the EVE.

[6] Orienting response is a series of physiological responses like pupillary dilation indicating increased attention to the stimuli (Sokolov, 1963, cited in Schmidt, 1991)

2.2.2 Primary Distinctiveness

Main characteristic of primary distinctiveness is its dependence on the context in which it is presented or perceived. There are two kinds: Perceptual and categorical distinctiveness. They come from either the item not carrying the same features with the rest of the group or not belonging to the category it is presented in. Another kind is external priority information, like an instruction to pay attention. Lastly, consistency effects, is whether the item does fit into the existing schema, framework or prediction of the subjects. This last kind is one that has most relevance to EVE. It was one of the main reasons offered by Hirshman (1988) that EVE was a result of the violation of the subjects' expectations or predictions about the nature of the pairs in the list.[7]

2.2.3 Secondary Distinctiveness

The main characteristic of secondary distinctiveness is its independence from the context in which it is presented. The two kinds are studies on unusual faces and orthographically atypical words. Another kind is generation effects which suggest that self-generated items are better remembered than externally provided items. Lastly, another kind of this category is bizarre imagery. A bizarre mental image created by a sentence like the 'the girl bit the doll on the cheek', which creates better recall than a normal sentence, like: 'The girl kissed the doll on the cheek.' Bizarreness effect includes a surprise factor, that is, if the subjects are told about the

[7] The most well-known example of this kind of primary distinctiveness is von Restorff effect. Isolation effect studies in the literature actually started with von Restorff. In 1933, von Restorff has found that in a list of items of same category, if there is an item which belongs to a different category, this isolated item is much better remembered by the subjects than any other item in the list. Later this finding got renown as von Restorff effect (cited in Hunt, 1995).

bizarre sentences beforehand, the effect does disappear (Hirshman, Whelley and Palij, 1989). This surprise factor does not seem to utilize any assimilation as Schmidt (1991) explains that while bizarreness effect does not enhance integration of the new information to the already existing information, it does increase access to the memory.

2.2.4 Processing Distinctiveness

The fourth category Schmidt (1991) mentions is on tasks which is assumed to lead to different processes or varying levels of distinctiveness in memory traces. This category contains studies which are linked to distinctiveness yet cannot be placed into the three categories mentioned in the above sections. The studies here enhance memory through both between- and within-subjects designs. Also study-test congruence, as in depth of processing studies, where the depth of processing does not necessarily lead to better memory but the equivalence of depth in both study and test conditions lead to better memory, are important (Morris, Bransford and Franks, 1977).

Overall, all these categories are artificially divided and most of the processes mentioned have an aspect or attribute, which can be placed in any of these categories. The same can be applied to EVE too. Hirshman (1988) states it has a surprise response akin to orienting responses in the emotional distinctiveness category. EVE also could be placed in the primary distinctiveness category simply because the effect is observed only with within subjects design. Likewise, even though a little far-flung, EVE also has some connection to the fourth category, because the surprise response seems to play a crucial role in triggering the right strategy of recall. But, Schmidt (1991) categorized EVE as related phenomena to secondary distinctiveness, Hirshman and colleagues (1989) categorized EVE within the family of Bizarreness Effect, which is also a member of secondary distinctiveness. The idea here was to review what kinds of general processes and categories of isolation effects are there, which may be of some relevance to EVE.

2.3 Expectation-Violation Effect (EVE)

A subcategory of isolation effects, EVE has first been reported by Hirshman (1988). EVE represents the better retrievability of the weaker associated word pairs in comparison to strongly associated word pairs. This section first describes Hirshman's (1988, 1989) studies and later describes Amster and colleagues' (1992) study.

2.3.1 Hirshman's Studies

Hirshman (1988) reported results which are in contrast to the LAS. In a free recall paradigm, Hirshman studied the effects of the presence of weakly related pairs within a list of strongly related pairs, on the overall recall rate. The procedure consisted of a group of subjects seeing 19 pairs of words on a screen, pair by pair, for 10 seconds each. Subjects were told there would be a memory test. Subjects wrote the pairs down. After the presentation of the experimental list, subjects received a distractor task of visual word search for five minutes and later, a free recall for three minutes. They had a six item practice list[8] and one minute cued recall of that list in the beginning of the experiment.

In Hirshman's (1988) study, the findings showed that under certain conditions, weak pairs were much better recalled than strong pairs. Hirshman called this finding: The Expectation-Violation Effect. In his paper, he confirmed the existence of EVE with 13 different experiments. The different experiments were intended to narrow down the boundaries creating EVE.

Following is a list the basic findings of these experiments: EVE is established with *free recall*, not with cued recall or recognition.[9] This means that the subjects needed to be using their own retrieval cues and their own search strategies. If an external cue, even the items themselves were presented during retrieval, EVE disappeared. So, EVE had to be relying on *retrieval cues* generated by the subjects and these needed to be powerful enough to provide the advantage to the weak pairs.

Next, EVE is established only in *within-subjects design* i.e., where the weak and the strong pairs are presented together to the same subjects. Also, there has to be a *ratio* where weak pairs are fewer and strong pairs are more numerous in the list. Hirshman (1988) found this proportional difference to be critical. If the proportions were reversed or changed otherwise, the effect disappeared.[10] The need for these lists to have a certain proportion of strong and weak pairs suggests that the weaker pairs must *stand-out* on a background of stronger items.

[8] Practice lists are applied to eliminate the intervening factors due to subjects' getting acquainted with the task.

[9] Recognition: A retrieval test where the item is presented to the subjects. In recall, it is suggested that there is a search phase, where the subject has to search the semantic memory for the item-to-retrieved. In recognition the search phase is assumed to be skipped as the item-to-be retrieved is already being encoded. Subjects only need to verify what they see matches their memory.

[10] Specifically, 4 weak and 12 strong pairs were utilized in Hirshman's (1988) experiment and in Amster and colleagues' (1992) experiment.

EVE occurs with the recall of both response and stimulus words, in addition to complete pairs. From this, Hirshman (1988) concluded that EVE is both an encoding and a retrieval issue affecting the *representation* of the word pair.

Hirshman (1988) suggested that the last two critical findings are necessary, though not sufficient to obtain EVE. They are the necessity of committing more *blind-alley searches* for the weak pairs and the utilization/interference of the "resultant surprise response (called a *blind-alley search cue*) to cue the retrieval of weakly related pairs that are associated with the blind-alley search cue" (p.55).

Blind-alley searches are failed search attempts to find a semantically meaningful association between the words of a pair. Hirshman (1988) suggested that just like finding relations between pairs does improve memory, failing to find a relation can also improve memory, because the blind-alley searches represent more extensive searches being done within the semantic network. The more blind-alley searches are committed the more surprise response, i.e. blind-alley search cue can become associated with the weaker pairs and consequently create a retrieval cue. Hirshman suggested the beneficial processing received by the weak pairs must be a result of blind-alley searches. In an attempt to study the significance of blind-alley searches, Hirshman used homographs[11] in two types of pairs with dominant or non-dominant meaning associates. His reasoning was, if the non-dominant meaning of the stimulus word is associated with the response, it should produce more searches for the same response word, than when compared with a dominant meaning.[12] The

[11] Homographs are each of two or more words spelled the same but having different meanings and origins.

[12] For example, 'bug-insect' and 'bug-anger' would be dominant and non-dominant pairs respectively.

pairs had equal associative strengths, so that number of the blind-alley searches was the only remaining variable. He found stronger EVE with non-dominant pairs, suggesting that number of searches is critical for EVE. In a later experiment, he equated number of searches and found no EVE; again he was lead to the same conclusion.

Blind-alley search cue utilization during retrieval is the last critical conclusion made from Hirshman's (1988) study. He claimed that during retrieval weak and strong pairs are in response competition, i.e., *interference*. To test this, Hirshman improved the encoding of strong pairs by increasing their general contextual cues. He provided a temporal segregation of the strong pairs by presenting an extra blank slide (10 s.) between blocks of strongly related and weakly related pairs. This manipulation led EVE to disappear, and the stronger pairs were recalled better than in other within-subjects design experiments he has conducted on EVE. His conclusion was that retrieval-interference between strong and weak pairs was effective: while weak pairs have facilitated recall from their search cues, strong pairs' standard utilization of general contextual cues is overshadowed by the blind-alley search cue.

Hirshman's Bizarreness Study

Hirshman, Wheely and Palij (1989) did a more extensive study on EVE to find out more about its conditions. Considering bizarreness to be causing a similar surprise response, this time, they used the same basic experimental settings for the recall of bizarre sentences. Their purpose was to observe how surprise could be acting on the memory.

The material of Hirshman (1988) study contained pairs of words like: 'Rug-Carpet', 'Rug-Smelly'. The material for Hirshman, Wheely and Palij (1989) had sentences like: 'the girl kissed the doll on the cheek' or 'the girl bit the doll on the cheek.' They used these sentences with the ratio of isolated vs. normal items as in Hirshman's 1988 study. This time the list contained 4 bizarre sentences to 12 normal sentences.

Hirshman (1989) found that EVE and bizarre-imagery were affected in the same way by the same experimental manipulations. The effect was applicable only with free recall conditions and the proper bizarre/normal ratio again. Their conclusion on this study was that EVE and bizarre-imagery effects were of same kind of phenomena. Bizarreness effect did include a surprise factor. Because, the effect disappeared when the lists got longer, indicating habituation, or when the subjects were informed beforehand on the presence of the bizarre items. "Specifically, bizarre sentences are better remembered than normal sentences because surprise responses to bizarre sentences increase the association of items in bizarre sentences to the general contextual cues" (Hirshman, 1989, p. 594). Thus, the surprise response was responsible for increased association between the bizarre items and the general contextual cues, either through repeated processing or orienting response. Either way, this was a naturally occurring effect, not being aided by imaginal encoding instructions. So, they preferred to rename this phenomenon bizarre-context effect.

As was mentioned in the review of isolation effects, bizarreness effects were categorized as a member of secondary distinctiveness. In 1989, Hirshman and colleagues have found that expectation-violation and bizarreness effects are both members of same category as they both obeyed the same laws. These studies were a support for the blind-alley searches and the blind-alley search cues, i.e., surprise responses to be critically important in EVE. Even though Hirshman (1988) stated the critical importance of blind-alley searches and cues, he again suggested that they could not be sufficient if they were the only sources of information for retrieval. He wrote:

> Hunt & Elliot's (1980) description of the roles of semantic and nonsemantic cues in retrieval claims that nonsemantic information is effective only when used in conjunction with semantic information. This position predicts that the expectation-violation effect should disappear if the items in weakly related pairs are exceptionally weakly related. This is because subjects cannot semantically elaborate exceptionally weakly related pairs at study and the blind-alley search cue, a nonsemantic cue, is not sufficient to mediate the retrieval of exceptionally weakly related pairs in the absence of such elaborations. (p.56)

With this statement Hirshman (1988) suggested that exceptionally weakly related pairs, i.e. pairs which would be next to zero with respect to associability, would not be sufficient to create EVE. Because, they would not be able to utilize the semantic information enough and that nonsemantic cues dominating these pairs would not be sufficient.

2.3.2 Amster and Colleagues' Study

Hirshman (1988) states a study, which he has conducted, has supported his predictions that exceptionally weakly related pairs do fail to bring out EVE. The details of the experiment are not available in the article and the experiment does not appear to be published later either.

Amster and colleagues (1992) wanted to follow-up on this claim that if the pairs were exceptionally weakly related, EVE would fail to occur. Hirshman (1988) claimed that exceptionally weakly related pairs would fail to dominantly utilize semantic information thus, having mainly the non-semantic cues to work with as the dominant mediator for retrieval, EVE would disappear. Therefore, the goal of Amster and colleagues was to check whether the nonsemantic cue was a sufficient factor for EVE.

Associability Rating

In order to use exceptionally weakly related pairs which would allow for nonsemantic cues to be utilized dominantly and compare it with weak pairs, Amster and colleagues (1992) had to create triplets of pairs containing strong, weak, and exceptionally weakly associated pairs which they labeled as 'zero'. Below Table 2.1 gives an example of these pair triplets assembled from the ratings they have developed and used in the study of Amster and colleagues (1992).

Table 2.1: Sample pair triplets (Amster and colleagues, 1992)

STRONG		WEAK		ZERO	
Flight	Air	House	Air	Ruler	Air
Table	Chair	Pretty	Chair	Anger	Chair
Baby	Child	Money	Child	Butter	Child

Theoretically, there is no 'absolute lack of semantic association' as any concept could eventually be associated to another concept, no matter how farfetched they are. So, ultimately, there is no 'zero' association strength pair. For practical purposes though, the 'lack of semantic association' is approximated by a rating in association norms, which is the pair mostly rated as having association strength close to none.[13]

Amster and colleagues (1992) developed new set of associability ratings to have a pool of word pairs to be used for the experimental lists. Lists of word pairs included all the pairs from the studies of Hirshman (1988) and also were selected from Bilodau and Howell (1965), Palermo and Jenkins (1964b) and Keppel and Strand's (1970) association norms to make the triplets. These pairs were placed in lists to be rated by 252 undergraduate students. The students rated these pairs of words on a seven-point scale, with zero being 'not at all associated' and six being 'very strongly associated.' So, using all the words from Hirshman's study in addition to other words from various association norms, a new list of pairs was pooled, allowing the older pairs within the list to be updated with the current strengths of association of the time and society which the experimental subjects too came from. That is, the strengths of association in the ratings and the experimental subjects in Amster and colleagues were all students of University of Texas at Arlington who were enrolled in classes in 1992.

From these associability ratings, Amster and colleagues (1992) picked out the words rating of 0 to .5 associability as zero strength pairs, 2 to 3.5 point ratings as weakly associated pairs, and 5 to 6 point ratings as strongly associated pairs. Hirshman's weak pairs, on the other hand, were lower than the strong pairs on average. His strong pairs were about 5.01 and weak pairs were about 3.33 on a 7-point scale.

[13] For this study, pairs from the extreme low end of the scale which would still enable the experimenters to have the necessary amount of zero pairs to be used in the experiment were selected. And that was from 0 to 0.5 rating on a 7-point scale.

Design and Procedure

Amster and colleagues (1992) partially replicated[14] Hirshman's (1988) study. They applied the same tests and boundary conditions, except for the presentation of the weak pairs: Hirshman had a single *weak* pair group, whereas Amster and colleagues had a group of *weaker* pairs, which were composed of weak and zero pairs. This change from the original method was done in order to test for Hirshman's insufficiency of nonsemantic cue hypothesis, suggesting EVE should disappear with the exceptionally weakly associated pairs, i.e. the zero pairs of Amster and colleagues

Amster and colleagues (1992) divided the weak pairs condition of Hirshman's study into weak and zero association strength pairs. Each list was composed of two types of pairs: weaker and strong.[15] Weaker pairs were also divided into zero pairs and weak pairs, based on the selection criteria mentioned above. So, there were two types of lists, one that contained zero and strong pairs, and the other contained weak and strong pairs. There were two independent variables: Type of Pair[16]: a within-list manipulation of item strength, intended to test EVE; and Type of List[17]: a between-list manipulation of item strength. The dependent variable was the number of response words recalled from the experimental sets.

---

[14] Replication: "The conduct of an additional study in which the method of the first (usually an experiment) is precisely repeated. The term is sometimes used to indicate that the results of the second experiment confirmed the first, although this is a confusion of confirmation and replication (repeating). In confirming, one repeats and obtains the same findings. Similar, but different uses occur in statistics." (McGuigan, 1990, p.372)

[15] For a list of pairs used in the experiment with their association ratings and standard deviations and sample experimental lists, please see Appendix A.

[16] Type of Pair refers to the "strength of the pair association". They are combination of strong pairs and either zero or weak pairs, which are grouped under "weaker" label.

[17] Type of List refers to the type of list based on which of the weaker pairs it contains. They can be containing either zero pairs or weak pairs in combination with strong pairs.

After the presentation of the instructions, which included minding the subjects of an upcoming memory test, each subject was given only one list for later free recall. Subjects received a practice list composed of three pairs with counterbalanced strengths. The lists were counterbalanced as zero/strong/zero pairs and as strong/zero/strong pairs. This counterbalancing was necessary to make sure that the study strategies developed by the subjects imposed by the varying strengths of the practice files did not produce a bias in any group of subjects. There was a 1 minute delay between the practice and experimental lists. During the presentation of the experimental lists, the word pairs appeared on the computer screen for 10 s. each pair. During this period the subjects wrote the pairs down on a paper pad and turned the page over. After the experimental list's presentation, the subjects received a visual puzzle as a distracter task for duration of 5 minutes. At the end, a free recall test was given asking the subjects to write down as many of the response words of the pairs as possible for the next 3 minutes. Subjects were debriefed before they left the experiment.

Results

The data were analyzed with a Two-factor mixed design: Repeated measures on one factor ANOVA, type of pair being the repeated measure. The results showed a strong significant interaction [$F(1,46)=15.52$, $p<.001$] between the type of strength in the lists and type of weaker strength pairs.

Table 2.2: Results of Hirshman (1988) and Amster and colleagues (1992) studies[18]

			Type of Pair	
			Strong	**Weaker**
Hirshman (1988)[19]			.24	.40
Amster and colleagues (1992)	Type of List	**Weak Pairs**	.28	.23
		Zero Pairs	.28	.43

The results showed no significant main effects[20], but an interaction[21] between factors. There was no significant main effect neither for type of pair nor for type of list. This indicates that, overall there was no main effect for expectation-violation effect. Also, the lists containing weak pairs and zero pairs did not significantly differ in their overall recall. That is, there was no main effect for processing type.

The LAS effect was obtained when the weaker pairs came from the list that contained the weak pairs. That is, when the subjects were presented with the weak-strong pairs list, the classical effect of LAS was observed. The in-between strength pairs which were called the ‘weak’ pairs did show the classical results and were recalled more poorly than the strongly associated pairs. EVE was observed when the weaker pairs came from a list that contained zero pairs. That is, when the subjects were given zero-strong pairs list, the recall for the zero pairs was significantly better than strongly associated pairs and they showed a strong EVE. While the strong pairs

[18] “Weaker” in this table represents the weak pairs if it comes from a list containing weak pairs and zero pairs if it comes from a list containing zero pairs

[19] These results of Hirshman (1988) are from his 12th experiment among 13 experiments which tests the proportion of weak and strong pairs. In his first experiment which simply establishes EVE provides a .23 to .34 values.

[20] Main effect in statistics refers to the influence of that specific independent variable alone on the dependent variable.

[21] Interaction in statistics refers to the combined influence of both independent variables on the dependent variable.

were recalled at a comparable rate in all experimental lists, the weaker pairs were recalled differentially. Zero pairs were recalled better than both strong and weak pairs. Otherwise, the results were same with the classical findings.

Overall, Amster and colleagues' (1992) study confirmed EVE with zero-strong pair lists by the zero pairs from that same list being recalled better than the strong pairs in the same list. Weak-strong pair lists showed results suggested by LAS, as the weaker pairs from that list were recalled worse than strong pairs. Hirshman (1988) has claimed that if the association strength of the weak pairs were exceptionally weak, EVE would disappear, because they could not utilize semantic information. However, in Amster and colleagues' study, EVE is obtained only with the exceptionally weak pairs, i.e. zero pairs. When the pairs come from what might be called medium range, of 2 to 3.5 over 0 to 6 scale, the weaker pairs showed recall rate in line with the expectations by LAS. So, this study has shown that EVE is obtained with the lowest association strength pairs used in the experimental setting. When Hirshman's results too are considered, EVE in both experiments of Amster and colleagues and Hirshman, seem to have similar degree of EVE, but in different conditions.

Yet there also seems as there is a discrepancy in the results between Amster and colleagues (1992) and Hirshman's (1998). The results from Hirshman (1988) and Amster and colleagues seem to show equivalent proportions of recall reflecting EVE, but under different conditions (See Table 2.2). This discrepancy is because in both experiments the weak pairs had a mean of 3 points associative strength[22] and this gives the impression that same degree of associative strengths have failed to

[22] Please note that the association strengths do not have an absolute scale.

produce EVE in Amster and colleagues' experiment and produced EVE in Hirshman's experiment. But, in Amster and colleagues' it was not the weak pairs which produced EVE, it was the zero pairs. Pulling the weak pairs down to a level where almost no one has rated a possible association in between, did not diminish EVE. The zero pairs which bring out EVE seem as they come from a scale of association strength which Hirshman has claimed would not produce EVE. Therefore, the results of Amster and colleagues are not consistent with the claimed results by Hirshman (1988). Consequently, in this thesis, the simulation of the model is based on Amster and colleagues' study and findings.

Conclusion

Amster and colleagues' (1992) conclusion about their study was that there was a clear distinction between classical results aligned with LAS and EVE which can be produced when the lower association strengths are really pulled down to almost none. Amster and colleagues have confirmed[23] the presence of EVE, the presence of EVE with the exceptionally weakly associated pairs, and disconfirmed Hirshman's claim that nonsemantic retrieval cue, i.e. exceptionally weakly associated pairs, would fail to produce EVE.

Amster and colleagues (1992) attributed these results to EVE's lack of dominant utilization of semantic information. The exceptionally weakly related pairs were there to check for the utilization of non-semantic retrieval cues. It is not possible to claim that there was absolutely no semantic information utilized by the

[23] Confirmation: "The process of subjecting a statement (hypothesis, theory, law, and so on) to empirical test. The consequences may be that the probability of the statement is decreased (disconfirmed, not supported) or increased (confirmed, supported). Distinguished from replication in that replication refers to the repetition of the methods of a scientific study" (McGuigan, 1990, p. 368).

zero pairs. As it was mentioned before, there is always some kind of association between concepts no matter how far-flung they are. Neither it is possible to claim that for the strong pairs there was only semantic information processing. Like all memory phenomena, the processes are highly intertwined and practically never mutually exclusive[24].

Amster and colleagues (1992) showed that EVE can be observed with exceptionally weak pairs, which are close to almost no semantic association. When the subjects are given mediocre weak pairs they exhibit recall rates predicted by LAS. These results suggested that the dominant utilization of nonsemantic information is important for EVE. The utilization of nonsemantic information is combined with the surprise response as a result of keeping on failing to find a semantic association against all effort. This has inspired the model's algorithm as explained in Chapter 4, although the exact implementation chosen in the model is not claimed to be the exact mechanism by which EVE works. Rather, it was chosen as a practical avenue to provide correlation with Amster and colleagues' results within the limitations and the constraints of the model and ACT-R.

Hirshman (1988) found that EVE emerges when certain conditions are met. These boundary conditions are the type of memory test applied (free recall), type of experimental design (within subjects) and the number of weakly related pairs in the study list (about 4 to 12). These mean that subjects rely on their own retrieval cues and organization to retrieve these pairs, these cues form as a result of some form of comparison between weaker and strong pairs and the weaker pairs stand-out on a background of strong pairs. Hirshman's (1989) study adds the information that

[24] That is one of the reasons why a 'dominant' usage is mentioned often.

surprise responses play a mediating role in establishing the memory for the isolated items, bizarre sentences in this case.

Hirshman (1988) suggests a theoretical explanation of EVE by suggesting that the failure to understand a relation between two items can improve memory performance for that word pair. He calls these failures blind-alley searches. Blind-alley searches occur when there is a novel and unexpected semantic combination and result in a memory representation as blind-alley search cue. The blind-alley search cues, or surprise responses as Hirshman suggested, mediate the retrieval of the pair later on with an additional benefit from attention.

As Amster and colleagues' (1992) study have found that EVE occurs at the lowest end of the associability scale, they have concluded that exceptionally weakly related pairs utilize blind-alley searches more dominantly than semantic processing. So this thesis makes an assumption that blind-alley searches and the resultant cues are critical to produce EVE.

CHAPTER 3

ACT-R as a Cognitive Architecture

ACT-R is a theory of human learning and cognition. It is a cognitive architecture for simulating and understanding basically all cognitive phenomena, like how people think, perceive, organize and use knowledge, and also produce intelligent behavior. The researchers in ACT-R strive to integrate the psychological findings into this cognitive architecture with the goal of getting ACT-R closer to performing the full range of cognitive tasks.

This chapter presents the ACT-R cognitive architecture. Since the purpose of this chapter is to give a basic understanding of it, the discussion proceeds without going deeply into matters that are not essential for the understanding of the model developed in this thesis. However, the assumptions of the ACT-R theory and the specific techniques used for the modeling that are directly relevant to the model are discussed in greater detail.

3.1 The History of ACT-R

Since 1973, ACT (Adaptive Control of Thought) got revised under the heading of many different versions. Main ones are HAM, ACT, ACT* and ACT-R. Anderson and Bower (1973) set the origins of ACT, when they established HAM as a theory of human memory. At the time, the influence of associationistic perspective in memory has become such a dominant idea that Anderson and Bower (1973) developed a general theory memory based on the associationistic theory, which they named Human Associative Memory (HAM). HAM was a description of human declarative memory with an associationistic approach. This model which is a predecessor of ACT-R, assumed a propositional network context where ideas are represented as nodes and the links between them are the relations, i.e. associations. All the semantic information is represented by the configuration of the links and the labels on them. The nodes themselves have no semantic labels. The activated nodes spread activation to the nearby nodes causing them an increase of activation in turn. During the retrieval, a probe or cue, matching a terminal node triggers an activation which follows the associations searching for the right concept, or node. When the activation reaches the sought-after item, if it is strong enough and if it did not fade out through fan, it can be retrieved. If the nodes are frequently activated contiguously the association between them grows stronger leading to faster retrieval times. HAM was a model of human memory, and not of human cognition. Thus, it was inevitable that revisions were needed to be able to account for many of the critical issues.

ACT-R's beginnings of being a unique theory of mind started with Anderson (1976, cited in Anderson and Lebiere, 1998) when he presented ACT with a significant change from HAM. ACT included an additional procedural memory. So now, ACT had both declarative and procedural memory within its design. While chunks were the basic units in declarative memory, the productions made the basic units of procedural memory. ACT with this basic design, was capable of computational adequacy to account for various phenomena of cognition and established the basic format of future ACT models. Anderson (1983, cited in Anderson and Lebiere, 1998) a newer version, ACT*, was a result of the realization simulation of mind is bounded by how the brain functions. So ACT* was the attempt to bring neurally plausible subsymbolic processing into the model. Another contribution of ACT* to the line of ACT models was the ‘theory of production rule learning’.

ACT-R is the latest version of ACT. It is inspired by yet another important point: human mind is an adaptive and ‘rational’ mechanism. An adaptive mechanism’s goal is to represent the outside world most efficiently and most truthfully. Thus, in ACT-R, where R stands for ‘rational’, the goal was to add rational analysis of cognition. The other purpose was to improve the details of the subsymbolic level of processing. As it refined the activation calculus, it also improved the production learning. ACT-R is also important for allowing productions to be conceivable as the atomic components of thought, i.e. really small, critical and basic units.

Another version of ACT series of architectures, which is noteworthy, is ACT-R/PM. The PM stands for Perceptual-Motor extension. It intends to situate the modeled cognizer in a realistic environment, not isolating perception or motor actions from the act of cognition.

3.2 ACT-R

The current state of ACT-R is described in the article *An Integrated theory of the Mind* (Anderson, Bothell, Byrne, Douglass, Lebiere and Qin, 2004). The basic architecture of ACT-R 5.0[25] consists of several modules, each processing different forms of information. The modules are coordinated by the central production system, which is not sensitive to the processes in these modules; rather it coordinates these modules through the module specific information contained in a buffer of each module. The central production system operates with respect to the contents of these buffers, and it also has the power to change their content. Some of the modules which are included in ACT-R and how they interact are represented in Figure 3.1.

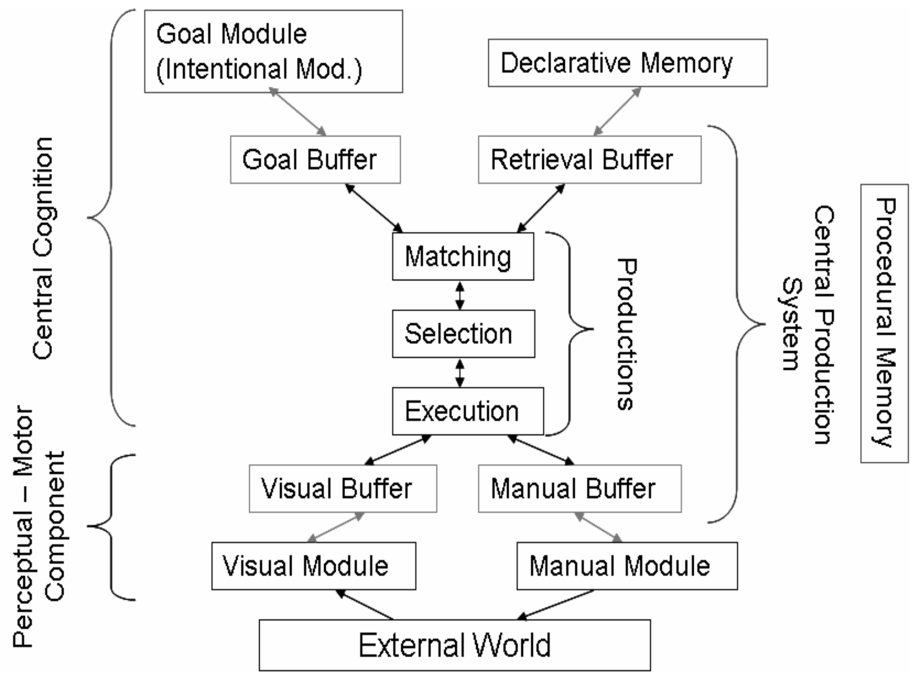

Figure 3.1: The main structure of ACT-R (Adapted from Anderson and colleagues, 2004)

25 ACT-R 5.0 was the current version of ACT-R at the time the model was being developed.

Historically, ACT-R first focused on central cognition, which is represented by Declarative Memory Module, Production rules (also referred as Central Production System and Procedural Memory), and Goal Module (also referred as Goal Stack in previous versions, and Intentional Module). These modules are discussed in the following section. Later, as was mentioned above, the perceptual-motor component was integrated, which implements the basic visual, auditory, vocal, and manual tasks. The division into central cognition and perceptual-motor system has nothing to do with the importance given to the modules or their complexity; it is just the matter of division of labor.

3.2.1 Central Cognition

Central cognitive component of the ACT-R architecture is made up of Declarative Memory Module, Procedural Memory Module and the Goal Module. These three systems represent the central cognition which performs the higher level processes of cognition. Declarative and Procedural memories relate to each other revolving around the current goal, which is determined by the goal module.

Procedural Memory

The procedural memory is considered to be the critically important part of ACT-R system. Procedural memory consists of productions, which are considered by Anderson (1998) to be the atomic components of thought. The information processing is performed by productions of the central production system or procedural memory.

Productions are if-then statements, which refer to the definition of a condition and the action that has to be taken in response to that condition. They suggest possible actions if certain conditions are present. The production system is composed of these, which are called production rules. The production system acquires and uses productions based on the syntax of the production rules. Which productions match to the conditions and which ones should be selected or executed are determined by utilizing symbolic and subsymbolic calculations.

An example of a production would be:

```
(p find-result
   =goal>
         isa          ADDITION-PROBLEM
         number1      =n1
         number2      =n2
         result       nil
         state        "find-result"
==>
  +retrieval>
         isa          ADDITION-FACT
         addend1      =n1
         addend2      =n2
  =goal>
         isa          ADDITION-PROBLEM
         state        "requesting-result"
)
```

In the above example the `=` sign preceding an expression denotes that the expression is a variable; thus, `=n1`, `=n2`, and `=goal` are variables. The `+` sign, on the other hand, specifies the requests made by the production to various modules. In this case, `+retrieval` specifies the retrieval request made by the production system to the declarative memory module. The `==>` sign is a divisor; it separates the condition part of the production rule from its action side, i.e. IF from THEN.

If we would translate the syntax of the production rule given above to more “English like” form, it would be read as follows:

IF the goal is a ADDITION-PROBLEM, and
the first number is =n1, and
the second number is =n2, and
our state in the problem is to find the result
THEN
request the retrieval of the
ADDITION-FACT chunk, whose
first addend is =n1, and
second addend in =n2
And note that the request for the result was made

There are few critical aspects of the production rules: The first one is their *modularity.* Modularity stands for their independence in activity; independence in the sense that production rules are learned separately, one at a time. They are the units of acquisition and deployment of knowledge. While the production rules are learned one at a time, based on single instances, they can go beyond those single cases by the process of *abstraction*. Abstraction makes use of variables, to be replaced by the object of the current situation/condition.

The system searches the right production to take care of the task at hand, i.e. current goal. Productions are condition-action rules that form the basic rule/guiding system in response to a certain situation. The selection of which production to apply is done through *conflict resolution*. Many productions are listed based on their match with the goal. The selected *productions* from the procedural memory have three options: they can produce an action, they can request further information from the declarative memory or they can transform the current goal.

An important difference between procedural memory and declarative memory, which will be described next, is in the *conditional asymmetry* of the procedural memory. Conditional asymmetry refers to the fact that when information is being searched, the direction that goes from condition to action can not be reversed. This aspect applies only to the procedural memory. For declarative memory, both in theory and in practice, the search can be started from either direction with success. As the procedural memory is a further compilation of the information to the point of remaining only with the rules/connections, it does not have the conscious resources of investigating the information.

The procedural memory has complex assumptions integrating the declarative chunks within. What this means is that, while the procedural memory is a pile of rules, it makes frequent use of declarative information. This calls for complex assumptions to be made by the procedural memory to open up the declarative chunks as needed. When chunks are further assembled, productions are created. This process is called *production compilation*.

Declarative Module

The main information processing module, i.e. central production system, utilizes the information store which is called the declarative memory. Declarative memory is a memory store in the classical sense, keeping facts about the world or ourselves. The basic units of declarative memory are chunks. Chunks are grouped pieces of information that we acquire as a result of learning. They are compact and independent, and they represent our knowledge of the world.

There are two origins of a chunk. Perceptions from the outside world create our *object chunks*. The end result of our learning through responses to the goals is our *goal chunks*, which are based on our experiences. “A chunk is defined by its type and its slots” (ACT-R Tutorial, unit 1); and these two kinds of chunks have slightly different structures. The type of object chunk is an object category chunk belongs to, and its slots can be seen as *attributes* a member of that category might have. In contrast, the type of goal chunks is a ‘type of pattern’ or relation, describing relevant information about a situation in the form of a goal, and its slots can be seen as *arguments*. But eventually, these two kinds of chunks are syntactically identical and are processed identically.

A typical example of a chunk is our knowledge of an addition task, such as 2+4=6. In ACT-R a chunk is represented in this manner:

```
isa ADDITION-FACT
        addend1     Two
        addend2     Four
        sum         Six
```

The type of this chunk is ADDITION-FACT, and its slots are arguments used in addition: addends and sum.

If a retrieval request to the declarative memory is made, the retrieved chunk is returned to the current goal, e.g. Addition-fact of 2+2=4. If the goal is completed with success, then it is returned to the declarative memory as learned information.

<u>*Goal Module*</u>

Another module, which the Central Production System constantly interacts with, is the Goal or Intention Module. The structure and nature of the goal module is under the consideration in the ACT-R community, but its function is clear. It allows the system to function in an organized fashion. Goal Module represents a hierarchy of goals or intentions people carry in order to choose how to deal with the external world and be rational about it. Thus, human ability to act differently to the same external state depends on the goal module. For example, if we are presented with certain numbers say, 36 and 64, whether we add, subtract, or divide them; or do completely some other kind of operation, like dialing the phone, does depend on the current goal. Although, there can be several goals, since they are ordered hierarchically and dealt one at a time, there can only be one current goal. ACT-R is a fixed-attention system that is, the whole system works towards the current goal, which is the focus of attention. However, this does not make ACT-R an inflexible system, because the current goal can be replaced with a more valued goal, if it ran into. This allows ACT-R to account for distractibility and opportunism observed in behavioral data.

As it was mentioned above, declarative memory chunks are divided into two – object chunk and goal chunk. It was also said that goal chunks are accomplished or completed goals. Also, in order to simulate some task, there has to be the representation of it, which is also sometimes referred to as problem space. This representation is the chunk that occupies the goal buffer at the beginning of that task, which is also the chunk in the goal buffer and is in the focus of attention. It is totally up to the modeler to specify the time of the creation of such chunk, and the type of the created chunk; that is, the number of slots the chunk has and the initial values of these slots. Since goal chunks are representation of the task, they are constructed in such a way that they contain all the information that is necessary for the current task to be carried out. The slots of these chunks are filled and modified through out the task. The way how slots of the goal chunk are manipulated is also up to the modeler. When the task is finished, i.e. the goal is accomplished; the chunk which keeps the information cumulated during the task is added to the declarative memory; and this chunk can be seen as representing the task. Then the system becomes ready to work on another goal or task. The actions of the system after accomplishment of the goal depend on the model, that is, it is again the modeler who determines what will be done next.

Since it is the modeler who specifies the structure of the goal chunk and the type information that is encoded in its slots, the accomplished goal chunks may carry different kinds of information like encodings from environmental context or personal experiences. Thus, in different ACT-R models such chunks are treated and referred to differently.

3.2.2 Perceptual/Motor Extension

The perceptual – motor aspect of ACT-R attempts to make ACT-R a more comprehensive theory of cognition. Rightfully complaining about the older trend in psychological theories separating cognition from perception and from action, Byrne and Anderson within Anderson and Lebiere (1998) develop ACT-R/PM. With this, they intend to establish a comprehensive theory of cognition. Citing the fact that, in reality, perception, action and cognition are not separable in a real sense, they stress the importance of having a comprehensive theory.

In ACT-R architecture, the information coming from the external world is taken into the perceptual buffers of the perceptual modules. The reason is that the buffers form the medium for the central cognition to interact with different modules, through the procedural system. Buffers allow the central cognition to take and/or revise information unitarily. The perceptual-motor component in ACT-R is an extension implemented to respond to ACT-R's goal to be a unified cognitive architecture.

The perceptual/motor system has several modules. These modules can take several commands from the cognition layer. The asynchronous nature of these modules allows for parallel processing. These modules and the central production system can operate in parallel: A task can start before another task is completed. There are vision, manual, speech and the audition modules defined by ACT-R/PM.

ACT-R/PM has not played a critical part in this model, thus detailed discussions on it will not be further informative.

3.2.3 The Processing and Subsymbolic Level in ACT-R

There are various levels of assumptions in ACT-R. At the goal level ACT-R can be considered as a symbolic and discrete system. At the procedural and declarative processing levels, the calculations are continuous and subsymbolic. There are two levels/places where parallel processing takes place: when an appropriate production is searched for and when an appropriate chunk is searched for.

At the subsymbolic level of ACT-R, parameters are used. The parameters are means to govern the behavior of the system at the subsymbolic level, and thus dictate the way of responding to the external state. They enable the model to simulate a rather independent behavior towards the external conditions, i.e. power of choice.

The subsymbolic part of ACT-R consists of the conflict resolution and chunk retrieval. Conflict resolution is related to procedural memory and refers to how the appropriate productions are selected. Chunk retrieval consists of declarative processing and refers to the information retrieval per productions request.

Conflict Resolution

The selection of productions to be applied among the many requires a *conflict resolution* protocol. There are two basic steps of conflict resolution: The first step is the procedural part. Using a comparison on the goal states, possible productions are put into a *conflict set*, i.e. all the production whose condition side's buffer tests match the current content of the buffers. Then they are ordered according to their expected gain. The expected gain is calculated by a simple calculation of probability of achieving the goal P_i multiplied by the value of the goal G. When the cost, as measured in time C_i is subtracted from this we have the net utility of the production U_i.

$$U_i = P_i G - C_i$$ **Production Utility Equation 3.1**

Declarative Memory Retrieval

Declarative memory retrieval is an important aspect of the model developed in this thesis. Consequently, it is discussed in more detail than other aspects of the ACT-R theory.

As was mentioned above only one chunk is retrieved at a time. Similar to the production selection process, the declarative memory chunks are retrieved with respect to some value. This value is the activation of the chunk A_i. It is calculated by the formula given below:

$$A_i = B_i + \sum_j W_j S_{ji} + \sum_k P_k M_{ki} + \varepsilon_1 + \varepsilon_2$$ **Activation Equation 3.2**

For the better perspective on how this equation works, every addend is discussed separately together with the equations that govern their values. After the discussion of the addends this equation will be summarized.

B_i is the base-level activation of the chunk. It reflects how much the chunk was used before. The value depends on three parameters - the time that passed since the chunk was used last t_j, total number of usages n, and the decay parameter d. The base level activation of a chunk is calculated by the formula:

$$B_i = \ln \left(\sum_{j=1}^{n} t_j^{-d}\right)$$

Base-Level Learning Equation 3.3

The decay parameter has the default value of 0.5 and it simply reflects the speed of forgetting. Every usage increases the base-level activation of the chunk; consequently, it is important what counts as a usage. They are also referred as presentation, practice, re-presentation, or reference in ACT-R theory. There are three cases: First, creation, second, merging, and last, retrieval. How chunks are created was mentioned above, they are accomplished goals or encodings from the environment. The creation of the chunk counts as its first usage. It was also mentioned that productions make retrieval requests to declarative memory. Successfully retrieved chunks are used by productions, and each retrieval adds a usage to a chunk. The merging occurs when the newly created chunk matches exactly with another chunk in the declarative memory (i.e. two chunks has the same slot values). In this case, instead of adding this new chunk to declarative memory, the chunk which is already in the declarative memory receives a presentation. Thus, there are no duplicate chunks in the memory, which is more efficient.

The second addend in the activation equation is referred to as associative activation, source activation, or contextual priming. As the name implies, it has to do with the association strength between the chunk to be retrieved and the chunks that are in the slots of the goal chunk. A chunk which is a slot value of the goal chunk, acts as a source of activation for the chunks it is associated with. For chunks, e.g. j and i, in order to be associated it is necessary to have the ia (interitem association) value (S_{ji}). If the chunk in the goal chunk is the slot value of the chunk to be retrieved, the S_{ji} value is calculated automatically. The S_{ji} value depends on the number of parameters as well, and calculated by the formula:

$$S_{ji} = S - \ln(fan)$$ **Strength of Association Equation 3.4**

In the formula S is the maximum associative strength a chunk can have (to itself, for instance), which can be set by ACT-R global parameter :mas. Fan, on the other hand, is the number of chunks associated with the chunk j. Consequently, the more associations a chunk has, the less its associative strength to any given chunk. However, the frequency of use of two chunks together, i.e. any chunk's being required when another chunk occupies one of the slots of the goal chunk, also increases the associative strength between two chunks. This process is referred to as associative learning.

Since the experiment simulated in this thesis does not focus on the life time learning of associations, but rather focuses on the effect that strength of association has on retrievability of extremely weakly associated pairs; and since the period of the experiment is too short to significantly affect the prior associative strengths, this process is not explained in further detail.

ACT-R makes it possible to manipulate the associative strength between chunks without manipulating these parameters. This is done to simulate the prior learning, i.e. the state after the long term associative learning has taken place. The ACT-R predefined command `set-ia` is used for this purpose, and values set by this command are used instead of the automatically calculated S_{ji} values.

The second parameter used in the associative activation calculation is W_j. It represents the importance given to that chunk, also called attentional weight. The amount of attention that can be given to a single chunk in a slot of the goal chunk is calculated by the simple formula:

$$W_j = W/n$$ **Source Activation Weighting 3.5**

Where W is the total amount of attention that can be distributed and n is the total number of chunks in the non-empty slots of the goal chunk (there can be empty slots). It is assumed that each chunk in the slots of the goal chunk receives equal amount of attention. The total amount of attention is also referred as goal activation, which also can be set by the ACT-R global parameter `:ga.` At the same time it is assumed that the W is the individual difference parameter, i.e. different people can have different W.

Total amount of source or associative activation is the summation over the products of W_j and S_{ji} for all chunks in the slots of the goal chunk. If the chunks in the goal slots do not have association or S_{ji} with the chunk to be retrieved, then the associative activation is zero.

The third addend in the activation equation is referred to as match score. If the partial matching is enabled in the model, the chunks that do not match the retrieval specifications exactly can be retrieved. The match score is 0, if all slots match, if there is a mismatch it becomes negative. In other words, this addend decreases the overall activation if the chunk somehow violates retrieval specifications. The partial matching is off in the model; consequently, the process is not discussed further.

Last two addends are permanent and transient noises. Their values are set with global `:pas` and `:ans` parameters respectively. The permanent activation noise is the one which is added when the chunk is created. However, the transient activation noise is a random value added each time the chunk is attempted to be retrieved. "The noise is a logistic function which is characterized by parameter **s** [`:ans` and `:pas`] which is related to a variance of noise distribution, ?^{2}, by function" (ACT-R 5.0 Tutorial, unit 6):

$$s = \frac{\sigma\sqrt{3}}{\pi}$$

Noise 3.6

To sum up, the activation of a chunk is the result of the frequency of its use, the contextual priming, the degree of match to the required specification, and the noises.

Besides the activation of the chunk, another global factor that affects the declarative memory retrieval is a retrieval threshold. It is a minimum value that the chunk activation must be in order it to be retrieved. It is said that the activation of a chunk must be above the retrieval threshold. Otherwise the retrieval failure will occur, i.e. technically, ACT-R will retrieve an 'error' chunk that has the activation equal to threshold, i.e. the error will be the chunk with the highest activation.

The activation of the chunk affects the speed of its retrieval. Basically, the higher the activation, the faster the chunk is retrieved. The retrieval latency is calculated by the formula:

$$Time_i = Fe^{-A}$$ **Retrieval Latency Equation 3.7**

In the formula `F` is the latency factor parameter. Its default value is 1, however, it can be manipulated to make all the retrievals faster or slower.

Other Declarative and Procedural Parameters Used in the Model

Besides the global parameters already discussed, ACT-R makes it possible to adjust parameters of specific chunks and productions. The model developed in this thesis makes use of this feature, and certain parameters were manipulated to adjust the model to reflect the state of subjects prior to experiment or to simulate the theoretical necessities, like a semantic network, whose modeling is beyond the scope and capacity of this thesis.

In order to reflect the prior experience of subjects, it is possible to increase the base levels of declarative memory chunks and set the association strength between them. The association strength manipulations are done by the `set-ia` function, as it was mentioned above. The base level activations are manipulated by setting the declarative parameters `:references` and `:creation-time`, which as their names imply, specify the prior usefulness and the time when the chunk was created respectively.

The only procedural parameter that was manipulated is the `:effort`, which basically sets the duration of the production execution. The default action time `:dat` is defined as 0.05 sec by ACT-R. However, sometimes it is necessary to make a production last longer. This feature is usually used to simulate processes that are difficult or even impossible to model by productions. This parameter too was used in the model of this thesis, since some processes were unrealistic to implement. The `:effort` parameter was used for the productions that represent certain processes which will be discussed in the next chapter.

Retrieval Procedures in ACT-R

An ordinary retrieval process in ACT-R depends on the activation levels of the chunks in the declarative memory. The chunk with the highest activation level, which is determined by the activation equation, is retrieved. The rest of the retrieval depends on how the modeler believes the processes should be implemented through the productions.

However, there are certain conventions on how free recall, cued recall or recognition is specified. As ACT-R productions follow the same logic with what might be happening in the information processing steps going on in a human subject's cognition. Thus, retrieval productions implementing these different retrieval processes implement the same steps that should be happening during their realization in human subjects.

In ACT-R technical terms, retrieval is putting a chunk that meets retrieval specifications, i.e. which has the same chunk type and slot values as specified in a retrieval request on a production rule's action side, into the retrieval buffer, so that production rule can access it. Whether a chunk will be retrieved or not, also depends on its activation, as was already discussed. The retrieval process is always the same. However the retrieval specifications will change with respect to whether it is a recognition, cued recall or free recall. For example, if there is a chunk of type item that has two slots `arg1` and `arg2`, whose values are `X` and `Y` respectively:

```
chunk1 isa item
      arg1        X
      arg2        Y
```

In recognition, a subject only has to judge whether the presented item was also presented during the study or not. In ACT-R terms, the retrieval specification will be like shown below:

```
+retrieval>
      isa         item
      arg1        X
      arg2        Y
```

In retrieval request, because the subject has full information about the to-be-retrieved item, the full structure of a chunk is specified with the type and all required slot values. The retrieval request is done with all the necessary information, and it depends on the activation level of the declarative chunk and other parameters like noise or retrieval threshold to allow for the retrieval of the item.

Although the example chunk contains only 2 slots and their values are both specified in a retrieval request, this does not imply that if a chunk contains more slots all their values must be specified. The above example is kept very simple for the sake of clarity. Continuing with the same example, in cued recall, where subjects are presented with only one argument, which functions as a *cue*, retrieval specification is looser. That is, the number of specified slot values is less.

In case of free recall, where subjects must retrieve items without being provided any cues, the retrieval specification will contain even less specified slot values or even none. In this case, the success of retrieval depends mainly on the base level activation of the chunk, and the associative strength with other chunks in the goal chunk that might function as a cue for it. Thus, the difference between recognition, cued recall and free recall with respect to ACT-R is in the number of slot values specified in a retrieval request.

Because of ACT-R's activation calculus, there is another difference. Normally, presented cues are encoded into the goal chunk, that is, they become a slot value of the chunk in goal buffer, before the retrieval request is made. Since chunks in the goal slots act as sources of activations, the same chunk will have different activation levels in recognition, cued recall and free recall. In recognition, it will be highest and in free recall, it will be the lowest. Because, in recognition there will be two, in cued recall one, and in free recall zero sources of activation, if the above example is considered.[26]

3.3 Models in ACT-R

For the ACT-R theory, like for the other cognitive architectures, it is important to be able to account for as many psychological phenomena as possible, because the greater the number of phenomena the theory can account for, the greater the possibility that it develops in the right direction. The accountability of the theory

[26] It is common for ACT-R models to add a 'context' slot to the chunk type specifications in such studies, because it is assumed that context information will make the activation of the required chunks higher; thus, filter out most of irrelevant chunks. Consequently, it is technically more accurate to say that in this case recognition will contain two more sources of activation than free recall, and one more than cued recall.

is expressed in the number of different experiments whose models were constructed, and whose simulations produced the results close to the results obtained on the real experiment.

Though there are various phenomena in psychology literature which are simulated in ACT-R, none are directly related to isolation effects or EVE. Isolation effects is an area which happens to be ignored by ACT-R modelers. ACT-R related studies and ACT-R models cover fields such as: Perception and attention (Anderson, Matessa and Lebiere, 1997), learning and memory (Anderson, Bothell, Lebiere and Matessa, 1998; Anderson, Fincham and Douglass, 1999), problem solving and decision making (Anderson and Douglass, 2001; Gunzelmann and Anderson, 2003), categorization (Anderson and Betz, 2001); language processing (Anderson, Budiu and Reder, 2001; Budiu and Anderson, 2004) and various other fields[27].

A simple example of a model which implements paired-associate learning is a simulation of the study by Anderson (1981), which is presented in the ACT-R tutorial, is comparable with the model in this thesis with respect to utilizing paired associates. Anderson reported an experiment in which subjects studied and recalled a list of 20 paired associates for 8 trials. The paired associates consisted of 20 nouns like ‘house’ and associated digits from 0 to 9. Each digit was used as a response twice. During retrieval, the subjects saw the nouns and were asked to report the digit that corresponded with them. This study is used as an example of how base-level learning through presentations, i.e. retrievals and encodings, increases base level activation and makes a chunk more available, i.e. increases its retrievability.

[27] These are only sample citations from these domains.

In the model of Anderson (1981) experiment, only one chunk type was used, since digits and words were not represented as declarative memory chunks, for simplification; rather associations were formed between their visual representations, the actual written marks. And this chunk represents an association and a memory for the encounter with such a situation at the same time.

Although this model is comparable in certain areas with the model in this thesis, with respect to utilizing paired associates, it was not used as a base for the model of this thesis. There are a number of reasons for not using it as the base for building up the current model. First, the model of the experiment of Anderson (1981) is not concerned with associative strengths and their effect on the retrievability of pairs. Consequently, its chunk types and productions are not designed to allow processes the model in this thesis has pursued. However, the idea of using goal chunks as a memory for experience was considered in developing the model of this thesis.

Although there are certain parallels and similarities between ACT-R models, the requirements and demands of each new model is different. Because all the models pursue different aspects of the psychological phenomena, like investigation of decay function in paired-associate learning studied by Pavlik and Anderson (2003), they are often not compatible to be used as a basis, they can only be inspired from.

Another study which has some relevance to this thesis is the study by Anderson, Bothell, Lebiere and Matessa (1998). In that study, they explore the paradigm of list memory. List memory paradigm might be studied by serial memory, cued recall, free recall, recognition and implicit memory tests. The authors have attempted to provide an integrated account for the whole domain of list memory. Besides the use of standard ACT-R subsymbolic calculations like time-based decay and partial matching; they developed a unitary representation for lists to be used in all kinds of experiments.

Anderson and colleagues (1998) assume that "a list is organized as a set of groups and each group is represented as a set of items" (p. 347). That is, that there is a chunk for each group that encodes its position in a list and the size of the group and a chunk for each item that encodes its position in the group. Also they assume that in addition to retrieving chunks representing elements of the list, it is also necessary to retrieve chunks representing groups themselves. Thus, there must be production rules that allow this.

Although in the experiment simulated in this thesis there are also lists presented to subjects, while Anderson and colleagues' (1998) model mainly presents individual items, the experiment of Amster and colleagues (1992) uses lists of pairs. Even though one of the modeled experiments uses lists of pairs in a free recall paradigm, the focuses of interest in the compared studies are different: while Amster and colleagues are focused on the associative strength and its influence of recall, Anderson and colleagues focus on serial position effects, i.e. primacy and recency effects, in serial recall, free recall, recognition and implicit memory, without a consideration of associative strengths. Still, the model in this thesis utilizes the basic algorithms utilized generally in similar tasks, without adopting other models as its base.

The representation used in the model of this thesis is similar and inspired from the representation developed by Anderson and colleagues (1998). There are also two levels of representation, but they are different. Since Amster and colleagues (1998) presented items in pairs, in which case pair functions as a natural group of size two, the chunk type that represents group does not contain size information. Moreover, since the order of pairs is not important in the experiment modeled in this thesis, there is no information about the position of the pair in the list. Similarly, chunk type that encodes items is different; however, it can still be seen as following the positional encoding assumption. Since the focus of the model developed in this thesis is expectation-violation, the designed structures also contain expectation information.

Another idea that was inspired from Anderson and colleagues (1998) is the use of context. In their representation list functions as a context for groups, and groups function as a context for items. Similarly, in this thesis, experiment was used as a context for pairs, and pairs were used as a context for words.

Although theoretical assumptions of Anderson and colleagues are not violated, since the model of Anderson and colleagues (1998) was developed on the prior version of ACT-R, and makes use of features that were deprecated in the current version; building the EVE model upon it is quite implausible. Also, transferring their model into current version of ACT-R might as well require changes in representation and parameters.

CHAPTER 4

A Model for Expectation-Violation Effect in ACT-R

There are certain critical points about building a cognitive model. Taatgen (2005b) suggests selection of the phenomenon and then gathering of information followed by a task analysis and defining the steps of the process. After selection of a cognitive architecture the gathered information needs to be specified on this architecture. Later, the fitting of the parameters in order to match the predictions of the model to the psychological data needs to be done. Last is the evaluation of the performance.

Former chapters have presented the psychological phenomena this thesis is focused on, the relevant information and also described the cognitive architecture to be worked with. This chapter will present the algorithm of the model with the parameters being manipulated and later evaluate it. Section 4.1, of this chapter presents a basic algorithm of the model in a general sense. Section 4.2 describes the data presentation with the chunk-types and association strengths used, and how these were significant in the modeling manipulations. Section 4.3 again reviews the algorithm of the model, but with more in depth explanations of the processes and productions. Section 4.4 reviews the results and evaluates the model. Also, a full listing of the model code is presented in the Appendix C.

In this thesis the experiment conducted by Amster and Özyörük (within Amster and colleagues, 1992) was implemented on ACT-R cognitive architecture. The model includes the original experimental stages, like presentation, reading and encoding of the pairs in the study section, performing the distractor task, and a free-recall of these encoded pairs in the test section.

4.1 Basic Algorithm[28]

A review of Chapter 2 gives us an idea of what kinds of processes might be important in EVE. Findings of Amster and colleagues (1992) suggested that EVE was a phenomenon which occurred when the weaker association strength pairs were as weak as possible, i.e. zero. Lesser difference in association strengths did lead to the classical results in line with LAS. Also, Hirshman's (1988, 1989) studies have shown that in EVE, blind-alley searches were critically important. Another important factor was the surprise responses, which occurred as a result of repeated failures to find a semantically meaningful association and in turn were used as retrieval cues.

As can be seen from the Figure 4.1, when a subject (or a virtual subject for that matter) enters the experimental room, it can be assumed that s/he has three basic information: that this is an experiment on memory, they will be studying some information and because there is no indication otherwise they do not expect some bizarre information to be presented. Later 19 pairs of words appear on the screen for 10 s each. When a pair is presented on the screen, ACT-R or the virtual subject sees that there is a pair of words; and his/her task is to read and learn this pair.

[28] The ACT-R implementation of the processes mentioned in this algorithm can be found in Section 4.3.

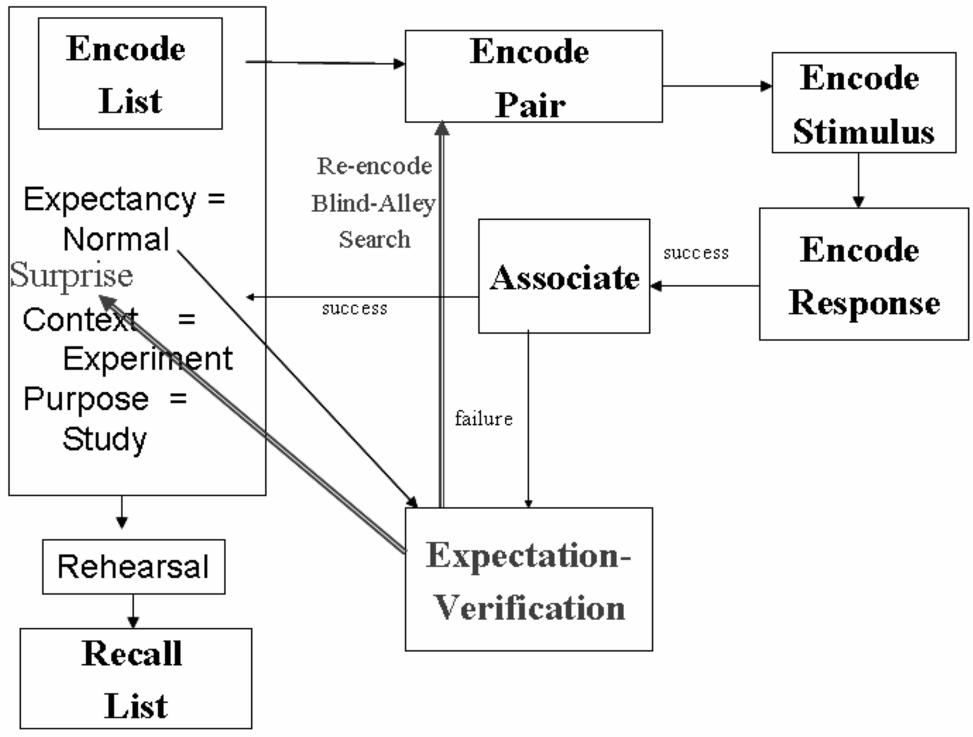

Figure 4.1: Simplified Algorithm for EVE Processes.

S/he proceeds with learning starting with the stimulus word, which is the word on the left, and then reads the response word, which is the word on the right. As these words have been used frequently before the experiment during daily life, they are already well learned and practiced. The subject successfully reads and understands the words, and, as the two words form a meaningful relation (with stronger pairs), s/he experiences no trouble in understanding the stimulus, the response and both of them together as a pair. Then the subject spends the rest of the time rehearsing these words.[29] But, when the pairs are extraordinarily weakly related, they do not form an easily understandable meaningful relation. Thus, the subject experiences some

[29] Subjects choice of rote or elaborate rehearsal, or even use of mnemonics is not simulated in this model. As a semantic network too has not been implemented, the complicacies of what happens during rehearsal is simply implemented by base-level increases to the pair chunks as described in Chapter 3 on ACT-R and will be explained in Section 4.3.3.

trouble when trying to read and understand this pair. With some extra effort spent in understanding these extraordinarily weakly related pairs[30], the subject reads and understands them successfully and starts rehearsing them just like s/he did rehearse the stronger pairs. This experience creates a surprise response in the subject, as s/he did not expect to see any pairs in the list which would be out of the ordinary.

When a new pair is presented on the screen the subject now has a new purpose or goal to read and understand this pair. So, the process described above repeats itself for the new pairs too. When the list is finished, the subject receives a distractor task: a real subject solves a puzzle for 5 minutes; however, in ACT-R, the passage of time is calculated.[31] After this, a free recall test begins, in which the subjects are supposed to remember as many response words as they can from the list they have just studied. So, the subject starts to report as many of the response words s/he can remember from the lists. Meanwhile, they remember having seen some pairs which were surprising to them, so, when they start remembering they start with these words first. This leads them to remember and report the extraordinarily weakly related pairs before the stronger pairs. Consequently the weakly related pairs are recalled more than the strong pair when these pairs are coming from the list which contains the extraordinarily related pairs, i.e. zero pairs.

When the subject reports all the pairs s/he can remember for the three minute retrieval time given by the experimenter, a new subject is taken to the experiment

[30] This section is implemented with the expectation-verification production shown in the simplified model algorithm schema in Figure 4.1, which will be explained in detail in Section 4.3.3.

[31] In the ACT-R environment this is simulated by calculating the decay which should occur during this time.

and is given the same procedure. There are a total of 48 subjects[32] equally distributed among the four lists, two of which contain zero association strength pairs as the weaker pairs of the list, and the other two lists contain weak association strength pairs as the weaker pairs, as it was in Amster and colleagues (1992) experiment.

4.2 Data Representation

This section talks about the representations in this model, within the context of the standard data representations used in ACT-R.

EVE is an exceptional case with respect to the regular findings of paired associate studies. EVE occurs with certain pairs when regular LAS results are expected to occur. There are certain factors which seem to lead these exceptionally weakly associated pairs to end with EVE.

This model utilizes certain manipulations to implement these factors for the attainment of EVE as a result. Hirshman (1988) stated that EVE is also a 'representational' issue. In order to be able manipulate the zero pairs which are to exhibit EVE and the strong and weak pairs which are to exhibit LAS results, this model implemented two different chunk-types for all the pairs within the experimental lists. These are pair and goal chunk types.[33]

The second manipulation affects the processing and it provides two forms of increasing base-level activation: one is through regular rehearsal and the other is through re-presentation–as blind-alley searches. An algorithm for the model was set

[32] There are several ACT-R parameters (i.e. settable values that affect performance of the model) that can simulate intersubject and within subject variation. They are discussed in relevant sections in this chapter.

[33] For a general review of these chunks please see Section 3.2.1. For more information on these chunks as how they are used in this model, please see the upcoming Sections 4.2.1 and 4.2.2.

such that with zero pairs the ‘goal’ chunk-types are dominantly processed, and with weak and strong pairs the ‘pair’ chunk-types are dominantly processed. That is, for zero pairs, goal chunk-types received most of the increase in the base-level activation, and for the weak and strong pairs, the pair chunk-type received most of the increase in base-level activation. This was a computational necessity in order to establish a distinction between processing preferences so that EVE can be observed.

Many memory models utilize chunk types similar to the pair chunks as representation of semantic memory processing. This model additionally used the goal chunks, which provides more contextual information then semantic pair chunks in order to simulate the blind-alley searches and the blind-alley search cues.

4.2.1 Chunk-Types

To implement the algorithm mentioned in a simplified form in Section 4.1.2 total of three chunk-types were used. The algorithm required a pair-level analysis, for encoding and associating the pairs as a unit and a word-level analysis, for processing the lexical meaning of the words. Also, as with every ACT-R model, there was a goal chunk-type as the focus of attention, encoding the current and relevant information within its slots.

Word Chunk-Type

At the implementation level, this first chunk-type, i.e. chunk-type `meaning`, was devoted to the lexical level representation of the ‘word’ for the subjects’ processing of the meaning of the word. Although this chunk does not contain any information that could be counted as a meaning of the word, its structure is sufficient to simulate this level of processing.[34] With the slot `word`, it represents the word as strings of letters. The slot `context` represents the connection of the word to a specific pair. When the pair chunk is formed, it receives a random chunk name. Through this random chunk name which is registered in the `context` slot, the words are assigned to a pair chunk. The `role` slot indicates the role this word is carrying, whether it is a stimulus or a response word.

```
(chunk-type meaning
      word
      context
      role
)
```

Creation of the `meaning` chunks has a benefit for these already well-learned words. The activation levels of the words begin and stay high throughout the experiment, and allow no unrealistic decay to occur on them.

The chunk-types, which were used for the implementation of higher-level analysis of pairs, were pair and goal chunk-types. As it can be remembered from Section 3.2.1., the `pair` chunk-type refers, as an ACT-R convention, to the semantic meaning of the pair and the `goal` chunk-type refers to the goal chunk which develops as the pair is processed.

Pair Chunk-Type

The chunk-type `pair` refers to the pair level and represents the meaningful association between these two words. Thus, it represents a higher level of analysis, utilizing the pair as a meaningful unit; even though, this chunk-type does not contain any semantic information within. Practically, it functions as a context for both

[34] This is usual practice in ACT-R models that does not focus on linguistic processing to represent the meaning of the word with a chunk that contains only a slot representing its spelling.

stimulus and response word; thus, it is encoded in the `context` slot of the meaning chunks. Thus, stimulus and response words can be seen as connected through these pair chunks. The significance of the pair chunk lies in its representation of semantic meaning; and it is utilized in this model to implement the ordinary semantic information processing on any memory task.

```
(chunk-type pair
      expect
      context
)
```

This chunk type has a slot `expect` for assigning an expectation value to the pair. It represents the subject's evaluation of the expectation for the pair. This value is initially set to 'normal', and it is switched to 'surprise' if EVE processes take over. Its second slot `context` represents the pair being studied in the context of an experiment. This is computationally required to allow the activation to spread through the list.

Goal Chunk-Type

As it was mentioned in Section 3.2.1, goal chunks represent the task, and determine the current goal of the system. As the system faces a new task a new goal chunk is created and it is added to declarative memory when the task is over. Since these chunks represent the problem space, they carry the relevant information regarding the current situation. For example, in this model the goal chunk is called `remember`.[35] It is devoted to be used as the 'current' goal of performing the necessary task of learning and reporting the pairs during recall within the

[35] This name is only informative and does not influence the functioning of the model.

experimental environment. Thus, it contains all the information necessary for the task at once.

The slots `stim` and `resp` are for assigning the stimulus and the response words. The slot `pair` is for assigning random chunk name, which is randomly assigned by ACT-R to register which pair was the encoded one. It also has a slot `expect`, for assigning the expectation of the subject on the difficulty of the pairs to be presented in the experiment, just like it is in the pair chunk-type.[36]

```
(chunk-type remember
        stim
        resp
        pair
        expect
        context
        purpose
        state
)
```

Goal chunk type `remember` also has a slot called `context`, which is used for encoding in what context the pair was seen: here, it is the experiment. The other slot of the goal chunk is called is `purpose.` It can take two values: study or test. With respect to whether it is test or study, different productions are selected for the same state. The last slot, `state` allows the modeler to control the flow of the productions as indicated in the algorithm. Figure 4.2 below gives an example of a goal chunk with all its slots filled with relevant information:

[36] The difference between expectation encoded in the goal chunk and pair chunk is in that in the goal chunk it is a difficulty subject expects, in the pair chunk, however, it is the difficulty status for the pair with respect to expectation for the list.

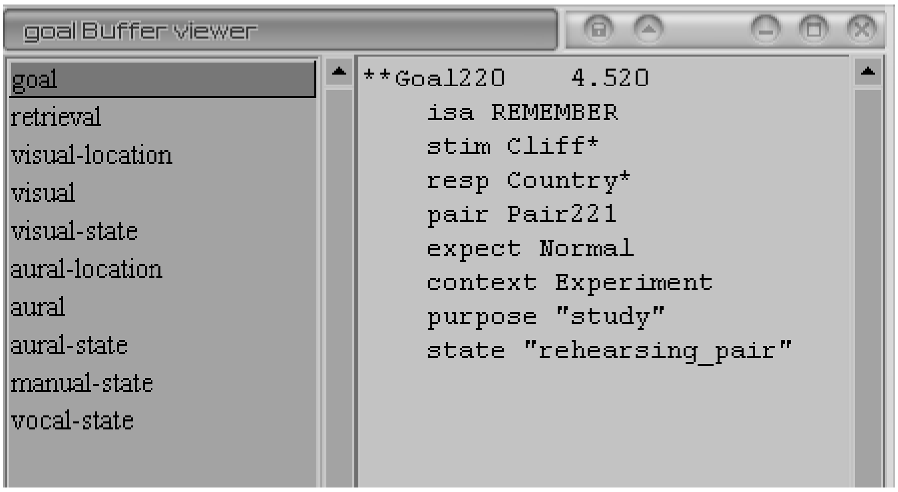

Figure 4.2: Goal Chunk-Type Remember

Once the productions run through so that this goal is accomplished, it is added to the declarative memory as a goal chunk with all the information registered with it. Consequently, this chunk happens to carry the information from that experience of encoding and processing the pair. It carries information regarding whether the pair was normal or surprising, or whether it was last studied or retrieved, and such. The choice of these slots depends on the demands of the task and the preferences of the modeler.

The chunk-type `remember` of the model in this thesis acts as the repository of the complete memory for the pair. In this model, the goal-chunk types encoded not only the words and the pairs, but also that the studied pair was surprising with a surprise tag, which results from the surprise response as a reaction to the extra effort required by the blind-alley searches.

In the model of this thesis, the goal chunk-type was also utilized for the implementation of the blind-alley searches and blind-alley search cues, i.e. surprise responses. First of all, goal chunk is needed for the representation of the problem space. Second, Amster and colleagues (1992) stated that non-semantic cues can be utilized in EVE. Since pair chunk represents the semantic memory for the pair, it cannot be used as a non-semantic cue. Thus, another chunk type was necessary for this purpose. The use of a goal chunks is a good option, since they were created for all the pairs, just like the pair chunk-types.

During the implementation of the algorithm the zero pairs, unlike the stronger pairs, went through blind-alley searches more than they went through rehearsals. Since blind-alley searches are offered as the reasons for EVE, and as EVE is suggested to be a representational issue, the assignment of a different chunk-type not only implemented a different representation, but also clearly separated the processes which these pairs go through, bringing computational clarity to the model. In short, it was practically much easier and cleaner to implement EVE on a different chunk-type. There are many factors suggested to be influencing EVE. Through Goal chunk-types, these factors like extra processing effort or extra attention, which these pairs receive, can be implemented and the information can be assigned to them without making extra assumptions. This provides a computational distinction. Consequently, allowing goal chunk-types to be a focus allowed for a much more handleable or easy to manipulate model.

Possibly a model which manipulated only pair chunk-types could have accomplished the same task, as bng as it could differentially direct them toward regular rehearsals or blind-alley searches. But this would only cause unnecessary complication on the part of the model's code. It is more realistic to implement EVE with goal chunk-types because EVE not only reflects some influence of experience like having to continually search for a meaningful relation or increased attention due to this extra effort spent on the pair, but also reflects the increased association to the context information, which are all registered by the goal chunk. Goal chunks provide a structure which is closer to the experience of going through blind-alley searches, especially when viewed from the perspective of ACT-R theory. This provides a more realistic approach in chunk-type selection, which blends with ACT-R theory.

Thus, it was not only computationally practical, but also more realistic to utilize a different representation for the pairs which go through blind alley searches. The goal chunks were selected as the chunk-type to reflect this representation, because they, as ACT-R default, encode the instances with all the relevant information.

4.2.2 Experimental Lists: Structure and Associative Strengths

As was mentioned before, the words used in pairs of this experiment are common words frequently used in daily life. Also, these words have certain strengths of association between them. In the model, this well learnedness and the associative strength between words are also reflected, so that virtual subjects start the experiment with previo us knowledge of these words.

This section comprehensively reviews the association activation in ACT-R with respect to the model in this thesis and later explains the use of association strengths in the model

ACT-R Activation Equation with Respect to the Model

According to the activation equation of ACT-R, the activation of a chunk has two main parts: the base-level activation and the source activation; as can be seen from Figure 4.5 on the activation equation and the spread of activation. Base-level activation reflects the activation level due to the learning that has been achieved until that moment. The source activation depends on two factors: the attention paid to the goal, i.e. goal activation, and the associative strength between the chunk and other declarative chunks that are currently in the slots of the goal chunk. Goal activation is represented by `W`, and set by `:ga` parameter, and it has a certain value, which represents total amount of attention given to the goal. This value is equally divided among the slots of the goal chunk, which are currently assigned a value. The chunks in the slots of the goal chunk reflect the current focus of attention. As can be seen from the Figure 4.5 below the declarative memory chunk `cliff` receives goal activation (which is also called attentional weighting), as it is registered in the goal chunk. But the declarative memory chunk `country` does not receive any goal activation, as it is not registered in goal chunk yet. The declarative memory chunk `country` receives associative activation through the second parameter, i.e. `associative-strength` S_{ji}. If the total activation, which is associative activation summed with its base-level activation with respect to the formula above, reaches to a certain activation level which exceeds the retrieval threshold, then the chunk can be retrieved and registered in the `resp` slot of the goal chunk.

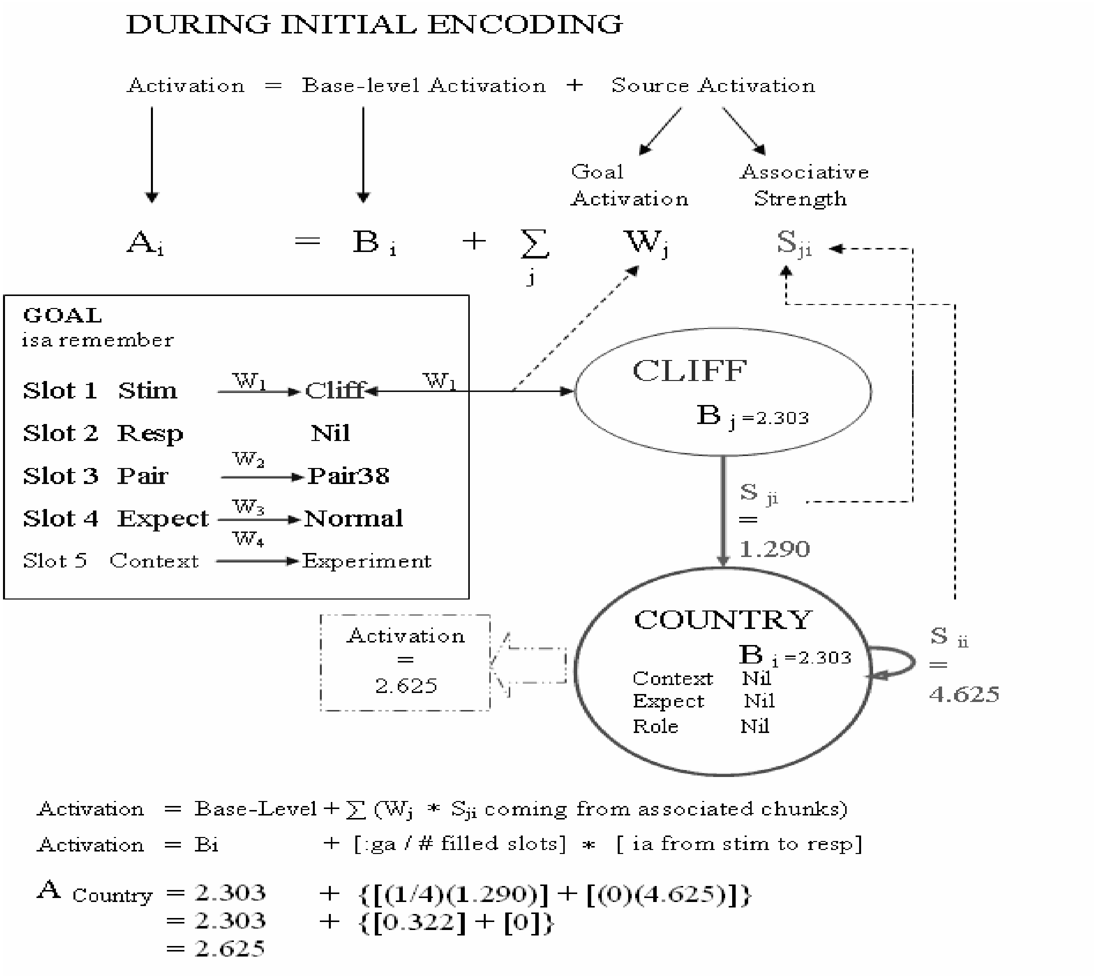

Figure 4.5: Schema for Activation Equation and Spread of Activation.

Base-Level Activation and Word Chunks The words used in the pair lists in this model are ordinary words, frequently encountered and well learned by anybody. So, the subjects in this experiment were assumed to have learned these words well before they came to the laboratory. Consequently, in the model, base-level activation representing the learnedness of the given items was manipulated with declarative parameters setting the word chunks to a well learned state by command (`sdp :references 500 :creation-time -10000`).

Source Activation: Attention With respect to the attention paid to the items, all the pairs and words are expected to receive about the same attentional levels. Therefore, the changes in attention towards the current goal when dealing with strongly associated items should not change drastically, and be around the default values. Consequently, in this model the `W` parameter was left at default values.

Source Activation: Associations to the Context These are basically the strengths of connections between items in the declarative memory. These association strengths were specified before the beginning of the experiment. When the experiment starts, the word pairs predetermined in the lists were presented by the `present-list` function. In the experiment, the pairs of words were presented only once to each subject. Also the words in these pairs occurred only in one pair each; thus, were presented only once to a subject. As the pairs are treated and encoded one-at-a-time, and also that this is not a semantic network simulation, it is sufficient for the model to provide associations for the direction from stimulus to response word only.

Associative Strengths

The experimental lists and the association strengths defined in these lists are very important in this model. As was mentioned before, they were taken form the original lists used in Amster and colleagues' (1992) experiment. Besides the association strength, these lists specify their order of presentation, and the experimental significances (strong, weak, zero, etc) of the pairs.[37] The directional (from stimulus to response) association strengths are used in the calculations of latency and effort value, which will be described in detail in Section 4.3.

[37] Please see Appendix A for sample experimental lists and their associative values.

There are two places where the associative strength is used. The first place is the latency of retrieval from declarative memory. In the model, this occurs during encoding and rehearsal as they are affected from the `latency`. Consequently, associative strength is one of the factors influencing encoding and rehearsal.[38]

The other place where the associative strengths are used is the calculation of the `effort` parameter of productions. The `effort` parameter stand for the processes affected by the associative strength and not implemented because of the absence of a semantic network. These processes are rehearsal and search for the association between two words, both of which are discussed in detail later on in this chapter.

The associative strengths used in the model are not realistic, because a semantic network has not been implemented. As it was mentioned in Chapter 3, in ACT-R it is possible to set them for each pair of chunks by `set-ia` command, and it is the way they were set in the model. How the associative strengths set by `set-ia` command are used and their effects are discussed later in this section.

Associative Activation and Retrieval Latency for the Words

During the encoding of an item, the perceived stimulus and response words are compared with the items in the subjects' memory. Thus, the stimulus and response words perceived on the screen are requested to be retrieved from the declarative memory, so that they can be encoded. The retrieval success and speed, during encoding or any other time when a request to the declarative memory is made, depends on their activation level. When the stimulus meaning is retrieved

[38] Associative strength is one of the parameters used in the activation equation, and activation of a chunk determines the chunk's retrieval time. How this values are used is discussed in later in this section.

from the declarative memory, and encoded into the goal chunk, it automatically starts spreading activation to the associated declarative memory chunks. The associative strength, i.e. S_{ji}, from stimulus to the response words influence the activation levels of the response words through priming. If the association is high, then the retrieval of the response word is primed positively, i.e. it takes less time to retrieve it; thus, the encoding of the response word is faster. Since in the model all `meaning` chunks representing the words, have the same base-level activation at the beginning of the experiment, the retrieval time of the response words from declarative memory depends on the activation they receive from the stimulus words.

The retrieval latency of a chunk from the declarative memory is defined in ACT-R by the formula: $RT = Fe^{-A}$. F is the latency factor (`:lf`), which is the model kept at its default value of 1. As latency factor (`:lf`) makes it possible to manipulate the retrieval times, it has an influence in the duration of encoding and rehearsal stages. It is possible to increase or decrease these durations by changing the latency factor. Since retrieval times reflect the priming effects, and since they depend on strengths of association and latency factor, these parameters are responsible for the priming effects on the response words.

Since latency factor is a constant for all pairs and all words have the same base-level activation, which is high enough to reflect the well-learnedness of these words and make them easily retrievable, the associative strength alone can make the difference in retrieval times between response words of different pairs.

It was mentioned above that the associative strengths of pairs taken directly from the study of Amster and colleagues (1992) and specified together with the experimental lists in the model. It was also mentioned that these values are used for calculations of effort values and to set the associations strengths, i.e. `ias`, between chunks representing words. Hirshman (1988) mentions 'failure of initial encoding' at the word level for weak pairs. Since the failure and the following success in encoding are supposed to take longer time than success alone, in ACT-R terms this theoretical requirement means that the response words of weak pairs should be retrieved slower than the response words of strong pairs. At the same time it means that the response words of weak pairs should be retrieved more slowly than the stimulus words of the same pair. Because of retrieval time calculations in ACT-R, these can be achieved only by making the overall activation level of the response word lower than the overall activation level of the stimulus word. Since the base-level activations of both the stimulus and the response words are the same, other addends of the activation equation have to be manipulated.

One of the possible addends that can be used for decreasing the overall activation level of the response word is the match score. It was mentioned in Chapter 3 that match score decreases overall activation of the chunk if it somehow violates retrieval specifications. Because in ACT-R activation calculations, the match score is the sum of the products of match scale, i.e. the importance given to match in specific slot, and match similarity. By default, the maximum similarity that chunk can have is 0, and the maximum difference is -1; consequently, mismatches produce the negative addend in the activation equation.[39] This feature was not used in the model because decreasing activation of the response word through this feature will require

[39] Although these values are settable; thus, can be both made positive or negative, using default values such that a mismatch would decrease the activation is more plausible.

simulation of the semantic network or, at least, very detailed representation for words.

Thus, the only other addend that can make the activation of the response word lower than the activation of the stimulus word is source or associative activation, which contains associative strength. Consequently, in order to satisfy the theoretical requirement mentioned by Hirshman (1988), the original associative strength used by Amster and colleagues (1992) were lowered by two points. The purpose of this reduction was to allow only the response words of the zero pairs to be retrieved more slowly than the stimulus words, so that at the word level the 'failure of initial encoding' can be obtained, which also reflects negative priming. The reason for choosing two points, specifically, is to allow associative strength of all zero and few 'weak' pairs become negative, producing the desired behavior. This was implemented when setting associative strength between declarative memory chunks by `set-ia` command. This manipulation only influenced the encoding and rehearsal stages, but not associating or blind-alley searches. Lowering these values has an effect on the overall results of the simulated experiment. However, because it stays at the word level processing, and as it was observed from the model trace[40] of the preliminary versions, this effect is not significant enough to be critical for EVE. But, it was still kept as such because it only enhanced the model.

The associative strengths were lowered only in `set-ia` command, not in the list declarations, because of computational reasons. In the list declarations the original values were used. The values in the declarations have to be positive since

[40] A trace in ACT-R model prints-out the requested parameter's effects on the model and the flow of the productions. Any level of detail can be chosen.

they are also used in calculations where negative result is unacceptable. These are `effort` value calculation, since it is the time spent during production execution, its value cannot be negative.

4.3 Main Processes and Productions

This section presents the virtual experiment on how EVE was implemented on ACT-R cognitive architecture. It describes what kinds of processes were implemented in this model. Also the productions implementing them, with the parameters utilized are explained.

4.3.1 Processing Stages

When any skill is new, it takes a lot of processing to perform. Later on, when the skill is repeated enough, it becomes automatic with the least amount of attentional resources devoted to it. Reading is a good example: first a naïve reader goes through the words, letter by letter; a more skilled one reads the sentences word by word; yet an expert reader skims a sentence focusing on a word or two in order to extract the information from a sentence. The shifts in focus and details do not only change in time over expertise, it also changes with our need on processing detail of certain items and objects. For example, reading newspaper headlines has a different focus than reading a philosophical argument. Depending on our expectations, the selective processes choose the objects to focus on as top-down processes. As Kahneman and Treisman (cited in Parasuraman and Davies, 1984) suggest, attention is utilized in top-down selection of objects to be processed, which are later processed at finer detail-level. This model took advantage of this flexibility of cognitive system in order to implement different levels of analyses. It was necessary to allow the model to make analyses both at lexical level and at a more global level of the 'pair'.

The model assumed a two-step processing style for the words and the pairs. First the 'pair-level' is processed and later the 'word-level' is processed. This was the implementational format of how processing was done during encoding, rehearsal and retrieval. All the pairs followed these steps at all stages. The two-steps representing different levels of processing are rationalized as skills that were learned before the experiment and have been proceduralized already. The word-level was treated as being integrated into the pair level.

Following are the reviews in detail of how the encoding, rehearsal and the retrieval were implemented in the model.

4.3.2 The Algorithm of the EVE Model

The schema depicting the detailed algorithm of the model with the relevant parameters is shown below in Figure 4.6. Following discussions of this chapter should be read in the light of this schema. As was mentioned earlier, the model's algorithm starts with the knowledge of being in a context of an experiment with the purpose of studying a list of word pairs, to be recalled later. As there is no indication otherwise, the expectation on this list is that it is a regular list with normal processing difficulty.

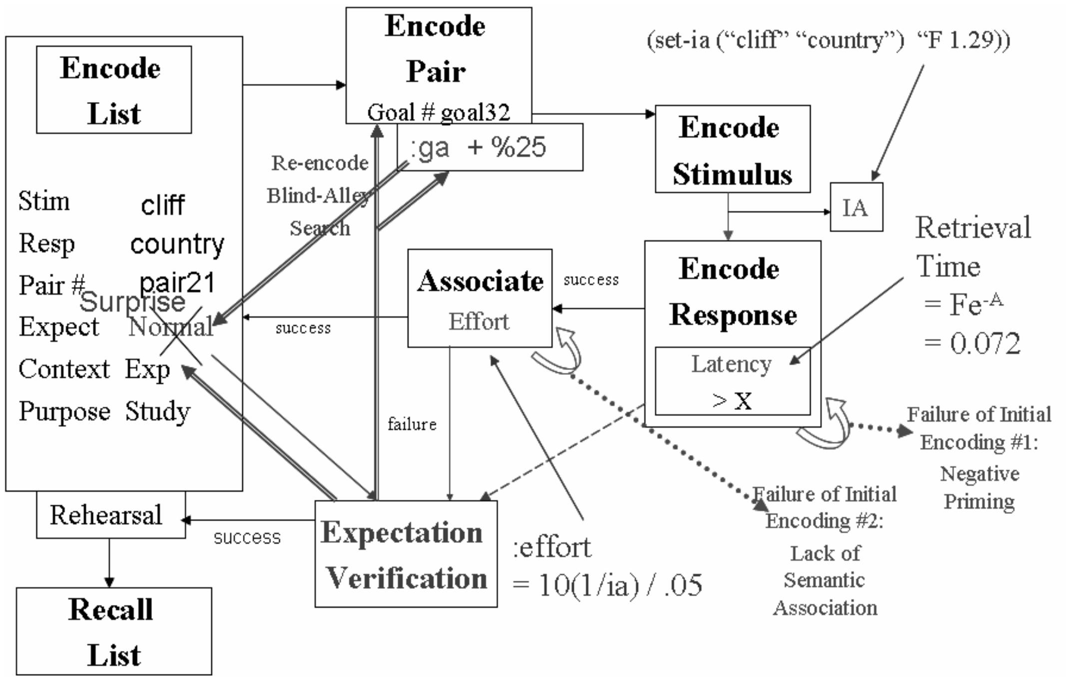

Figure 4.6: The Algorithm of EVE Model

4.3.3 Encoding

In the experiment, when the pair is presented on the screen a function makes the system process the screen and the attention is automatically drawn to the first word. Each pair is presented for 10 seconds.

The productions which simulate the encoding of the pairs start with the `attend-pair` production.[41] It acknowledges that this pair is being observed in an experimental context and that there is a 'normal' expectancy about the hardship of processing it. It also allows for a declarative pair chunk to be created and notes its chunk name which is randomly assigned by ACT-R. This random chunk name allows the pair to be tagged for identification.

[41] These productions are basic to ACT-R models and can be reviewed in model code in Appendix C. The productions which are critical to the model will be explained in full detail along with their code.

The next production `read-the-word` requests retrieval of the meaning of the text on the screen, being read as the result of the former production. The next production `encode-meaning-stim-and-find-resp`, encodes the studied word's meaning, that is, it retrieves the meaning of the attended word string from the declarative memory and assigns it to the `stim` slot of the goal chunk. It also assigns 'the stimulus' role to the first word of the pair. It also tags this word with the random chunk name of the pair it belongs to by registering stimulus word's context as the random chunk name of the pair. After that, it looks for the 'response' word, i.e. the second word of the pair. The same routine is applied for the response word. At the end of the production `encode-meaning-response` the goal chunk `remember` has all its slots filled with descriptions about the pair on the screen.

4.3.4 Rehearsal and Re-Presentation

After the words of the pair has both been retrieved and encoded, the model comes to the stage of associating these words. The `associate` production is a theoretical extension of re-attending to the pair as an evaluation of the semantic association between the stimulus and the response words. The hardship of associating the stimulus and response words is reflected in the duration of this production, i.e. its action or execution time, which is calculated and set as `:effort` parameter by the `present-list` function.

This stage of the model is a critical one for the implementation of EVE and provides ordinary rehearsal processes with an option for blind-alley searches, which was mentioned by Hirshman (1988), during the study phase. As can be remembered from Section 4.2 on data representation, there are two main manipulations in this model that were done in order to attain EVE. The first manipulation was representational, and it was discussed there. The other one is a process manipulation, which is the use of re-presentations in the processing of the experimental pairs in addition to regular rehearsals.

Hirshman stated that blind-alley searches were critical in EVE. Also, that they represent failed search attempts due to lack of apparent semantic association between the words of the pair. Accordingly, the algorithm of the model is set such that the zero pairs go through blind-alley searches. The strong and weak pairs skip this step of re-presentations, i.e. blind-alley searches, and move on to regular rehearsals; not because it is theoretically implied, but because it is computationally necessary.

The Associate Production

After word level of analysis is complete, the model turns back to the higher level analysis at the pair level. At this stage, having encoded both the stimulus and the response words, the model seeks a meaningful association between them. Consequently, the associate production is called after the encoding of the response word.

Associate production is a theoretical extension of evaluating the semantic association between stimulus and response words. As both the stimulus and response words are encoded, which are expressed with `stim =stim` and `resp =resp` on the condition side, the production calls `set-new-ref` function. This function increments the `:references` parameter of the goal chunks at each blind-alley loop they go through; thus, increases their base-level activation. Each newly added reference stands for the re-presentation, i.e. re-encoding of the goal chunk. Although the number of added references can be manipulated, in the model it is set to one. That is, the goal chunk receives only one re-presentation per blind-alley search. The total number of re-presentations received in blind-alley loops depends on the total `effort` value which has to be spent associating stimulus and response words; and this value depends on the strength of association between two words.

```
(p associate
      =goal>
            isa   remember
            stim  =stim
            resp  =resp
            state "re-attending-pair"
==>
      =goal>
            isa   remember
            state "rehearsing"
!eval!   (set-new-ref (goal-focus))
)
```

The `associate` production utilizes the `:effort` parameter to simulate the difficulty of establishing a semantically meaningful association between the words of the pair. Effort has equivalent meanings in ACT-R and common language. It reflects the effort which is spent in order to accomplish a task or to 'complete a production' in the case of ACT-R. In this model, it stands for the processing resources used by the subjects in trying to establish a semantically meaningful association between the words of the pair. It is inversely related to the association strength of the pair. That is, the more distant the pair is the longer it takes to associate. If the words are highly associated, then the `:effort` spent to associate them will be low. As the association of the words become weaker, the `:effort` spent in trying to establish the semantically meaningful association also grows in amount. How `:effort` parameter is related to the association strengths which are used in the model, are described in the subsection below. After this critical parameter's discussion, the thesis will continue with the explanation of the `expectation-verification` production, which together with the `associate` production implements the blind-alley searches.

The Effort Parameter

During the development of the model it was observed that the spreading activation alone and the retrieval latency from the declarative memory, unfortunately, were not sufficient to achieve the classical effect in ACT-R. The regular activation levels without any emphasis on them did not show enough effect to make the stronger pairs recalled better. Retrieval latency is an independent parameter, in the sense that it does not interfere or influence the activation equation. It only calculates how long the retrieval of the chunk from the declarative memory takes with respect to its activation level. As long as the retrieval latency and associative activation alone are taken into account, the only effect that can be obtained is priming. The spread of activation and the priming caused by it are not sufficient to produce an effect on memory that would be enough to lead to classical effect in an ACT-R environment. When the trace was observed, it was seen that the decay is stronger in ACT-R, enough to wipe out any influence or bias that may be produced by the associative strengths, i.e. `ias`, by the time of retrievals. This showed that another parameter is needed to make the associative strengths reflect or contribute to the better learnability of the pairs.

Solution came from a parameter of ACT-R, which stands for the time it takes to perform a production, `:effort`. `Effort` parameter changes the default production execution time[42] and makes the production take longer. In ACT-R terms `effort` is actually a value given to influence the production duration. And it is used in some ACT-R models to simulate learning new skills, skills that require more attentive and controlled processing.

In this model the `effort` parameter is manipulated by the `set-model-efforts` function. It calculates the effort values and sets them to be used by the `associate` production. In the formula, 1 is divided by the associative strength between the words to get an inverse relation, so that the effort value increases as the association strength decreases. There is a coefficient of 10 in order to bring the `:effort` values to an amount which makes them significant. Then it is multiplied by the production execution time of .05, because it is reasoned that every search in a semantic network can be seen as a production. Two examples of the most extreme pairs are represented in the below Table 4.1. The zero and the strong pairs represented in this table are the lowest and the highest association strength pairs used in the lists:

Table 4.1: The Effort Values with Respect to Pair Types.

	Zero	Weak	Strong
Effort = (10/ia) .05	7.14	.16	0.085
Ia =	.07	3	5.86

[42] ACT-R default is 0.05 s.

Effort is actualized in the `associate` production and `rehearse_pair` production during rehearsal. `Effort` represents the semantic searches, with an attempt to establish the association[43].

The Implementation of the Blind-Alley Searches

The implementation of the blind-alley searches was done through re-presentations or the re-encoding loops, which add presentations to the goal chunks and thus increase their base-level activation. Any pair can go through these loops. But, only the pairs which fail to associate *actually* go through these re-presentations. The details of which pairs and under what circumstances they go through these loops will be described when AEV (Associate-Expectation-Verification) loop is discussed.

The blind-alley searches are implemented as re-encodings or re-presentations. When combined with the representational manipulation mentioned in the previous section, the re-presentations apply to the goal chunks with every new search attempt as the item is being re-encoded.

The Expectation-Verification Production

The `expectation-verification` production takes over the role of initiating the blind-alley searches. While the stronger pairs pass through the `associate` production within a reasonable amount of time, the zero pairs fail to be associated during this time. But, the expectations are set such that the pairs should establish an association within a normal amount of time. Observing the contradiction between this failure and the expectancy of normal difficulty, the `expectation-`

[43] The failure of these searches are called Blind-Alley Searches.

verification production intervenes and orders a 're-encoding' of the pair.[44] This causes the goal chunk to be re-presented and increases its base-level activation, just like rehearsals increase base-level activation for the pair chunks. This 're-encoding' and the whole loop till back to the associate production represents the blind-alley searches mentioned by Hirshman (1988) and they will be described next.

As can be seen below, the expectation-verification production utilizes few functions to perform its task. After verifying that the state is normal and that there is still some effort left in the condition side, it calls for the set-model-efforts function in the action side, so that the pair's effort level will be calculated by ACT-R.

```
(p expectation-verification
      =goal>
            isa         remember
            state       "rehearsing"
            expect      normal
      !eval! (> *remain-effort* 0)
==>
      =goal>
            isa         remember
            state       "re-attending-pair"
      !eval! (increase-ga)
      !eval! (set-model-efforts)
```

Meanwhile, the expectation-verification production increases the attention paid to the goal chunk due to its prolonged demands in processing. This theoretical attentional increase was implemented by increasing goal activation (:ga) parameter, which directly increases the W_j values which were mentioned in activation equation, Figure 4.5.

[44] From this, it gets the name expectation-verification, it verifies whether the expectations are met. And if not, it takes action to make sure that they are.

The functioning of the `expectation-verification` production is integrated with the `associate` production through the AEV loop. Thus, it will be further discussed in the next section.

<u>The AEV Loop</u>

A closer look at the algorithm of this model describes the simulation of the blind-alley searches as represented by the Figure 4.7. The Associate-Expect-Verification (AEV) loop gives the general algorithm of how `associate`, `expectation-verification` and `review` productions interact with different types of pairs.

A criterion level, which is artificially set by the modeler, was used for representing the expectations of the subjects on the difficulty level of the processing of the pairs. This was selected to be a midpoint between the strongest zero pair's effort level and weakest weak pair's effort level, so that pairs, which the subjects have no difficulty in associating, can move onto rehearsals; and the pairs which fail to associate within a reasonable amount of time will be re-encoded until they do associate. Thus, the selection of this level was done to artificially separate pairs which exhibit ordinary results in line with LAS and the pairs which exhibit EVE, because the zero pairs needed to trigger the `expectation-verification` production in order to implement the 'expectation-violation' observed with them.

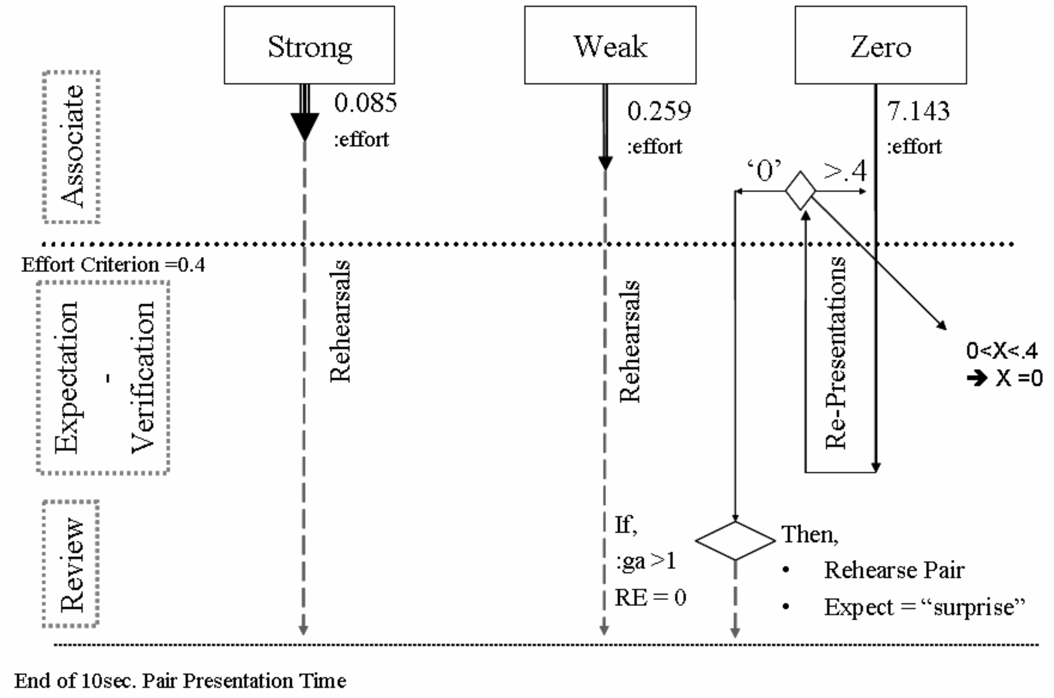

Figure 4.7: Associate - Expectation-Verification (AEV) Loop

As can be seen from the Figure 4.7, the Strong and Weak pairs establish a semantic association rather fast and they move on to regular rehearsals represented by the dashed lines. The `associate` production is a time-variable production, and that time is determined by the `effort` value. Thus, how fast the pairs will move on to rehearsal will be determined by their `effort` values. The pair-chunks of these pairs can spend the rest of the 10 s. duration of the pair presentation of the experiment, receiving rehearsals and with every rehearsal increasing the base-level activation of the pair chunk.

For the zero pairs, on the other hand, their `effort` value is much higher than the criterion. This represents the theoretical aspect of zero pairs failing to establish a semantically meaningful association within the expected amount of time. This reflects Hirshman’s (1988) statement on the initial encoding failure, at the pair level.

Computationally, this failure to associate within an expected amount of time has been implemented by functions calculating and setting the `effort` parameter at different times during this loop. These functions start with the `set-model-efforts` in the `associate` production.

The pairs which initially fail to resolve before the criterion trigger `expectation-verification` production. Theoretically, this includes any pair which has not yet established an association. Computationally, it is implemented by the remaining effort value being over 0, i.e. the production is triggered only by pairs whose remaining `effort` value is greater than zero, what happens when the `associate` production has not used up all the `effort` required to associate the words.[45] Within the `expectation-verification` production, as shown below, the `effort` value, which is used up during the `associate` production, is subtracted from the initial effort value and the remaining `effort` value is found. Then, the pair is sent back to the `associate` with the new `remaining effort` value.

This re-encoding loop continues until the amount of `effort`, which is calculated to take in order to find a semantic association, is spent (i.e. remaining `effort` becomes zero). At the end, when the pair is successfully associated, just like the stronger pairs, it does move on to regular rehearsals to be performed on this semantic connection. However, `Review` production, which is an extension of the `expectation-verification`, is triggered instead of the skipped `expectation-verification` production before the rehearsals. This production simply notes that

[45] It was mentioned before that the effort it takes to associate the pair is calculated before the pair is displayed.

the pair has gone through re-encoding loops and has high attentional level, and switches the expectancy or `expect` on the pair from normal to *surprise*.

```
(p   review
      =goal>
            isa          remember
            pair         =pair
            state        "rehearsing"
      =pair>
            isa          pair
         -  expect       surpise
      !eval! (zerop *remain-effort*) ;if remaining effort is 0
      !eval! (> (car (sgp :ga)) 1)   ;if :ga is greater than 1
      (criterion)
==>
      =pair>
            isa          pair
            expect       surprise
      =goal>
            isa          remember
            expect       surprise
            state        "rehearsing"
)
```

The critical point about this loop is that at each loop, a presentation is added to the goal chunk, which increases its base-level. Thus, the loop implements the blind-alley searches, which are searches in the semantic network starting anew every time, with the memory of the experience being strengthened every time.

Goal Activation Parameter

The theory of EVE and isolation effects suggests that attention is one of the important factors. Hirshman's (1988) suggestion that the surprise response may be explained in terms similar to orienting reflex was implemented as the attentional increase in the model of this thesis. Attention was added through the `expectation-verification` production with every re-presentation of the pairs. Each time the production was triggered, the goal activation was increased by 25 percent of the current. This number was selected arbitrarily, since it does not have an effect on the retrieval rate, i.e. the proportion of recalled pairs. As it was mentioned in above subsection, these increased attentional levels are used as a criterion for the selection of `review` production. Also, they affect the rehearsal of zero pairs, because increased `:ga` leads to higher source activation; thus, the retrieval time of declarative chunks is shorter. However, the difference in retrieval time is not great enough to affect the results produced by the model.

The encoded and associated pairs moved onto regular rehearsals where the pairs are kept on being successfully retrieved from the declarative memory. This was implemented by allowing the 'pair chunks' to move on to rehearsals, as they pass the `expectation-criterion`, and to be the ones receiving the presentation points, reflecting an increase in their base-level activation.

Summary of Rehearsal vs. Blind-Alley Searches

Rehearsal: A successful encoding and associating of the words of the pair leads subjects to rehearse them until another pair is presented. These pair associations are expected to be already established and subjects are expected to be just rehearsing them in their minds. Theoretically, during this stage the subjects are expected to be utilizing semantic elaboration mostly. However, since this model does not implement a semantic network, the semantic elaboration was simulated through `:effort` parameter. The time it takes to elaborate a pair was selected to be the same with time it takes to establish an association for stronger pairs, and the time of the successful search attempt in blind-alley searches for zero pairs. Computationally this implemented as the same `:effort` values for the `associate` and `rehearse_pair` production. In ACT-R addition of a presentation value to the pair-chunks, which in turn increases their base-level strength, leads to better retrievability.

Rehearsal in the model utilizes the same two-step manner as the implementation of encoding. First the pair is rehearsed, later the stimulus and the response words. Initiation of the rehearsal starts with requesting retrieval for the already studied pair. After the pair, stimulus and then the response words too are rehearsed. The rehearsal of the words is assumed to be subvocalization. Although ACT-R allows simulation of the subvocalization process by means of audition module, in the model subvocalization was simulated differently for the sake of simplicity. The :effort value of the production rules that simulate rehearsal of the words was set to be 0.35. The value is calculated with respect to the average length of the words used in the lists and ACT-R :syllable-rate parameter, which is the time it takes to speak average syllable and assumed to be 0.150 seconds. Since most of the words are bisyllabic, it would take 0.3 seconds to pronounce each of them. Also 0.05 seconds of default action time was added to consider the production that executes the subvocalization; thus, a word is rehearsed in 0.35 seconds.

Every rehearsal adds a presentation point to the pair chunk which increases the base-level activation for those pair chunks, and thus, increases their likelihood of being recalled later during the test session. The implementation of rehearsal of semantically associated pairs continues until the allowed time, which is 10 s. is spent. For this duration, the rehearsal process requests the retrieval of the same pair, i.e. the same pair is rehearsed for the 10 s. period. The reason for not allowing rehearsal borrowing is because there are too many factors on how the isolated items organize around or group during rehearsal (Rundus, 1971) and this is beyond the scope of this model as it intends to implement only the most basic algorithm possible. Regardless, there will be a brief mention of results with a preliminary attempt that allowed rehearsal borrowing, in Section 5.1.4. After that the 10 seconds is over, encoding of a new pair begins.

The *re-presentations* are different from rehearsals, because they actually represent a failure of encoding a semantic association and the consequent re-attempts in establishing that association. These are not literally semantic rehearsals, because the encoding of a semantic association has not been successful yet; they are re-encodings or re-presentations in ACT-R terminology. However, since all chunks in ACT-R have the same syntax and the same declarative parameters, technically, there is no difference between the presentation points received by pair chunks and re-presentation or re-encodings points received by goal chunks. All of them are expressed as `:reference` parameter. These simulate the blind-alley searches mentioned by Hirshman (1988), which continue until a semantically meaningful association is found.

4.3.5 Retrieval

As in any ACT-R retrieval process, in this model too, the pair with the highest activation level is retrieved. As can be remembered from Section 3.2.3, there is a convention on different types of retrievals and various parameters which are effective on retrieval of the chunks. This model implements free recall procedure and utilizes retrieval threshold and noise as manipulated parameters for retrieval.

Activation level of the declarative chunks is the main factor in retrieval in this model. Activation level of a chunk depends on many factors and this implies that, all the pairs that have been studied in this virtual experiment are not guaranteed to be retrieved. There are three factors affecting the variability in retrieval: main factor is the influence of the processes explained until now, which causes the pair and goal chunks of the pairs to have a variety of activation levels. Also, there is a temporary variation in the activation levels. This variation is created by the noise addend in the activation equation and is set by the :ans parameter in this model. This noise implements the variability of the human data. And the last factor is the retrieval threshold, which specifies the minimal activation needed by a chunk in order to be retrieved. For this reason, the retrieval of the pairs is not completely deterministic, and not all the studied pairs are retrieved.

Retrieval in the model intertwines the main processing manipulations, the representational manipulation and the processing manipulation, which was mentioned in Sections 4.2.1 and 4.3.4, and the output strategies of the subjects. The main processing manipulations were the representational manipulation of goal and pair chunk-types, and different processing stages of elaboration and blind-alley searches, which has been discussed extensively until now.

The model also implements the utilization of retrieval strategies mentioned in the literature about the isolated items, i.e. output priority, and the blind-alley search cue, i.e. the surprise response as a retrieval tag. In psychology research, it has been observed that distinctive items are utilized as retrieval strategies. In a list of distinctive and regular items which are presented together, the distinctive items not only tend to be recalled in groups (Bruce & Gaines, 1976; Schmidt, 1985), but they are also recalled earlier than the regular items (Schmidt, 1985). Bruce and Gaines also states that distinctive items are encoded into a smaller conceptual category than non-distinctive items and this increases their probability of recall. So, it is possible that distinctive items, the zero-association pairs in the case of Amster and colleagues (1992), are retrieved through a strategy which allows these pairs to be superior to strong-association pairs. As Hirshman (1988) too has stated, the retrieval strategies in EVE benefit from blind-alley search cue.

The output priority in retrieval is implemented by applying two strategies of retrieval: The first strategy, which is implemented first, is the retrieval request for the goal chunk-type `remember`. The second strategy, which is implemented next, is the retrieval request for the pair chunk-type `pair`. The first strategy implements the output priority observed for isolated items in general and which is also stated by Hirshman (1988). The second strategy implements the ordinary retrieval processes as what would be done if EVE was not observed and only results in line with LAS was observed. The first strategy calls for the memory of the surprise response, i.e. blind-alley search cue, with all the attended information encoded at that time. The second strategy calls for the semantic memory of the pairs, with the information that they were studied during the study phase of the experiment.

When the recall starts the first production[46] calls for the most active goal chunk, following the aim of the first strategy. When the goal chunk is requested, based on their activation level, the goal chunk with highest activation is retrieved.

[46] Please see the commented model code presented in the Appendix B for details of these productions which are quite lengthy, yet standard.

Thus, any of the goal chunks whether they belong to zero, weak or strong pairs can be retrieved. Because the zero pairs have the highest goal chunk activation due to their increased base-level activation from the re-presentations, i.e. blind-alley searches received, it is a zero pair which is recalled first.

Once the pair is retrieved, the expectation of the retrieved chunk is encoded into the current goal chunk. As the retrieved zero pair chunk contains the `surprise` tag, and now that it is registered in the current goal chunk, the activation starts spreading from the goal chunk to the declarative memory chunks which carry the same `surprise` tag. Because `surprise` pairs are fewer in numbers, the fan from the `surprise` tag is less, in comparison to `normal` tag in the `expect` slot of all other pairs. This leads to higher activations for the zero pairs.

The retrieval too in this model follows the two-step processing which was implemented during encoding and rehearsal. Thus, the second production of the retrieval, requests for the retrieval of the response word of the pair, which is already retrieved. Once the pair chunks are retrieved, whether goal or pair chunk-types, the response words are retrieved without any problem because the words themselves are not only tagged by the pair chunks, but also their activation levels allow them to easily pass the retrieval threshold.

The second retrieval production requests for the retrieval of the response words as Amster and colleagues (1992) have also asked for the retrieval of the response words. A direct retrieval request for the response words was not selected as an implementation, because it was not plausible for the observation of processing differences in EVE and LAS in this model. Neither theoretically it is completely correct, in the sense that the words are retrieved in the context of the pairs. Additionally, Hirshman (1988) states that EVE occurs independently of whether the stimulus, response or the pair itself is requested to be recalled.

When the model cannot retrieve the surprise tagged pairs any further, it switches to the ordinary retrieval strategies of retrieving the item chunks instead of the stored goal chunks. The second strategy calls for the `pair` chunk-types to be retrieved. Again the retrieval is based on the activation level. The same processes repeat with the `pair` chunks as happened with the `goal` chunks. This creates the LAS results in the model, which were also observed in Amster and colleagues (1992) results.

In the real experiment, the recall test starts right after the distractor task, which is used for the elimination of recency effects. In ACT-R this means that the words are not reported from goal or retrieval buffers where they can remain after the study phase; rather they are retrieved from declarative memory. Thus, the recall session in the model starts with clear retrieval and goal buffers, which are carrying no pair information from beforehand. Moreover, since distractor task lasts five minutes and the activation levels of chunks fall with time through decay, ACT-R also automatically calculates the new base-level activations resulting from the 5 min. decay.

After a `goal` or `pair` chunk is retrieved and encoded, it spreads activation to the associated stimulus and response words, which are retrieved and reported[47]

[47] Reporting was computationally implemented as addition of the retrieved word into the response list through Lisp functions.

next. Retrieved meaning chunks, as well as retrieved `goal` and `pair` chunks are tagged as `retrieved` so that they will not be attempted to be retrieved again.

After the retrieval and reporting of the response word, a new retrieval request for a pair is made. When there are no more retrievals to make, i.e. the retrieval is an `error`, the retrieval process is terminated. It can also be terminated brutally by the functions at the end of the three minutes recall period allowed by the experiment.

4.4 Evaluation of the Model

This section presents and discusses the results of the ACT-R simulation of the Expectation-Violation Effects reviewed in this thesis.

4.4.1 The Results

In this model the parameters for manipulating retrieval were noise (`ans`) and retrieval threshold (`rt`). `Ans` provides the variability of activation anytime a chunk is accessed, thus it provides noise. So, it is not as informative as retrieval threshold, which is an arbitrarily defined activation level forming a criterion for the activation level of the chunk retrieval. The chunks which are below this threshold will not be retrieved. Retrieval threshold is a cutoff point, by which the modeler decides an minimal level of retrieval, which s/he believes will be reflecting the processes being implemented.

The three different results presented below come from three different virtual experiments with 48 subjects each. As can be seen from the Table 4.2 below, the correlation varies from 0.999 to 0.964, with a mean deviation varying from 0.073 to 0.105, with different retrieval thresholds.

Table 4.2: Results of Virtual EVE Experiment

Retrieval Threshold		1.15		1.2		1.18	
		Type of Item					
		Weaker	Strong	Weaker	Strong	Weaker	Strong
Type of List	Zero	.50	.30	.42	.17	.46	.17
	Weak	.12	.21	.14	.21	.08	.19
Correlation		0.964		0.988		0.999	
Mean Deviation		0.073		0.082		0.105	

The Table 4.2 shows results of three different retrieval thresholds. These thresholds are close to each other, but having all three of them together presents more information at sight, about the effect being implemented. The retrieval threshold (`:rt`) 1.15, which is on the left of all three examples, has the lowest mean deviation and also the lowest correlation. As the threshold starts shifting from that location, the correlation increases to a fine level but, the mean deviation does increase too. Therefore, it seems like the correlation of 0.964 is the most prudent choice for accepting as a result of this study.

The results look promising that the model might have captured some basic trends in EVE. The basic idea of the model in this thesis, the manipulations intertwining processing and representation and setting up an algorithm which allows for different association strength pairs to respond differently might have captured EVE at the most crude manner; crude, because the many intricate theoretical issues have not been included in this model. For now, this exploratory work and its results bring promise for further explorations of EVE.

4.4.2 Limitations of the Model

One of the things that ACT-R assumes and bases its architecture on is an associative network representation. The model is a simulation of a process that goes through an associative network environment. But, associative networks are extremely subjective and there is no way to deterministically simulate them, except through building a semantic network. For any simulation strictly focusing on the strongly associated items or on classical effects as suggested by LAS, establishing a semantic network would be a reasonable way to study the details of the process. There is a lot of fuzziness when it comes to the defining, observing or manipulating semantic elaboration, semantic distance and such. An associative semantic network can pinpoint many of the questions which cannot be answered otherwise. Instead of establishing a network and simulating how the search is realistically accomplished, this research suffices with establishing parameters that can account for this theoretical aspect.

CHAPTER 5

Conclusion

In this thesis a computational model of Expectation-Violation Effects (EVE) has been developed, specifically on Amster and colleagues (1992) experiment. It is not claimed that the psychological processes are modeled through realistic means. Certain manipulations were done in order to implement EVE within the limits and constraints of ACT-R architecture. This is the first model of its kind simulating isolation effects through the subcategory named EVE.

The main contribution of this model has been to suggest how isolation effects and specifically, EVE can be studied through the modeling environment. The model is preliminary work, through which variations can be build upon for further developments, such as a more realistic implementation with a semantic network.

There are many cognitive architectures and ACT-R models describing attention, working memory, mental attention, learning, etc. But, until this time, there are no ACT-R models which simulate isolation effects or any effect that falls under any of the sub-categories of it. Isolation effects are an anomaly to our regular attention and memory functions. And just like any anomaly and memory phenomenon, any unified cognitive modeling architecture should be able to account for. A model like this one and its future versions, which simulates the field will be a contribution to the ACT-R literature as ACT-R strives to be more comprehensive.

5.1 Limitations and Future Development

As has been mentioned repeatedly, this study has merely been exploratory work. There is always more than one way of capturing human data. The finer distinctions on the issue need to be made at a theoretical level. This study helps bring out these areas. Future human experiments can benefit from different debates arising out of this preliminary work.

5.1.1 Blind-Alley Searches and Elaborations

As was mentioned before, an 'absolute zero' association strength can not be claimed. The blind-alley searches, which the zero pairs go through, do certainly follow a continuum from strong to the weakest pairs. In other words, with the weakest association strength pairs the subjects perform more of the searches and failing to find association and for the stronger pairs these searches are successful earlier on. The weaker the association strength of the pair is the more blind-alley searches are committed. Theoretically, the weakly related pairs too go through some blind-alley searches until their association is affirmed.

The model, focuses on implementing blind-alleys, which was suggested to be critical in EVE by Hirshman (1988). The model implemented this theoretical suggestion through a different strategy used on the zero pairs. The strong and weak pairs were implemented as going through regular processing. Consequently, the theoretical possibility of the few blind-alley searches committed in the weak pairs is not reflected in the implementation of the model. The model treats them as within the LAS group, through leaving them below the criteria of effort and letting them move on to rehearsal without going through blind-alley searches. This treatment is not intended to reflect theoretical and mutually exclusive processing styles by zero and stronger pairs. An improved version of this model should look into changing this arbitrary separation into a more continuous and natural one.

5.1.2 Expectation and Attentional Control

One of the possible avenues for future studies can be the contribution of attention in EVE, through computer modeling and psychological experimentation. A preexisting model of attentional processes, such as Norman and Shallice's (1986, cited in Baddeley, 1990) idea of Supervisory Attentional System could be a starting point for future development in this direction.

5.1.3 Semantic Network

Establishing a semantic network will be much beneficial for working on the details of the psychological processes and will be more fruitful when it comes to more conclusive results. It will be possible to focus in to the 'initial encoding failure' and 'blind-alley' search loops in detail. It will possibly also bring more clarity to the debate around semantic elaboration, distance, traversal and others.

5.1.4 Rehearsal Borrowing

Isolated items utilize distinctive or personal cues in retrieval, through output organization.[48] This output organization is a result of rehearsal strategies used by the subjects. During the study phase, as subjects tend to rehearse certain items together, they develop a subjective organization of the to-be-remembered material. The rehearsal strategy of subjects which groups certain items together, especially around the same category, leads to recall clustering. The organization determines their output priority during the retrieval phase. In short, the distinctive items are rehearsed together and recalled together (Rundus, 1971).

Unfortunately, there were a lot of variables in these rehearsal strategies which lead to the output priority. Like many other psychological processes, in this exploratory model, rehearsal too has been limited to the most basic simulation. The current model has implemented the end result of rehearsals of increasing base-level activation of what represents the semantic pairs with the theoretical implications of increased probability of recall for the pairs. No claims have been made on whether this reflects a rote rehearsal, semantic elaboration or any other means.

Preliminary runs were conducted to get a feel of what happens in this model, if rehearsal borrowing between pairs is allowed. When the rehearsal borrowing is allowed, through simply unrestricting the pair to be rehearsed, the correlation gets to about 0.99 to 1.0. Unfortunately, the mean deviation also increases so that it is not lower than 1. Thus, at this point the model does not seem to be ready for including rehearsal borrowing, without considering the variables influencing the output

[48] Organization is a process by which information is placed in memory in groups or it is rearranged in new and more optimal means.

organization. The future development of this model may pursue integrating these variables.

Overall, this model initiates the possibility for further investigations of isolation effects through the modeling environment.

REFERENCES

Amster, H., Brooks, C., Lucas, S., & Özyörükgee, N. (1992). The expectation-violation effect and the role of semantic relatedness in an episodic task. Bulletin of the Psychonomic Society, 30 (6), 463.

Anderson, J. R. (1976). Language, memory and thought. Hillsdale NJ: Lawrence Erlbaum Associates. In Anderson, J. R. & Lebiere, C. (Eds.) (1998). The atomic components of thought. Mahwah, NJ: Lawrence Erlbaum Associates, Inc.

Anderson, J. R.. (1981). Interference: The relationship between response latency and response accuracy. Journal of Experimental Psychology: Human Learning and Memory, 7, 326-343.

Anderson, J. R.. (1983). The architecture of cognition. Cambridge, MA: Harvard University Press. In Anderson, J. R. & Lebiere, C. (Eds.) (1998). The atomic components of thought. Mahwah, NJ: Lawrence Erlbaum Associates, Inc.

Anderson, J. R. & Betz, J. (2001). A hybrid model of categorization. Psychonomic Bulletin & Review, 8, 629-647.

Anderson, J. R. & Bothell, D. (2004). The ACT-R 5.0 Tutorial. Carnegie Mellon University, http://act-r.psy.cmu.edu/tutorials.

Anderson, J. R., Bothell, D., Byrne, M. D., Douglass, S., Lebiere, C., & Qin, Y. (2004). An integrated theory of the mind. Psychological Review, 111 (4), 1036-1060.

Anderson, J. R., Bothell, D., Lebiere, C. & Matessa, M. (1998). An integrated theory of list memory. Journal of Memory and Language, 38, 341-380.

Anderson, J. R. & Bower, G. H. (1973). Human associative memory. Washington, D.C.: Winston & Sons.

Anderson, J. R., Budiu, R. & Reder, L. M.. (2001). A theory of sentence memory as part of a general theory of memory. Journal of Memory and Language, 45, 337-367.

Anderson, J. R. & Douglass, S. (2001). Tower of Hanoi: Evidence for the Cost of Goal Retrieval. Journal of Experimental Psychology: Learning, Memory & Cognition, 27, (6).

Anderson, J. R., Fincham, J. M. & Douglass, S. (1999). Practice and retention: A unifying analysis. Journal of Experimental Psychology: Learning, Memory & Cognition, 25, 1120-1136.

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

Anderson, J. R., Matessa, M. & Lebiere, C. (1997). ACT-R: A theory of higher level cognition and its relation to visual attention. Human Computer Interaction, 12 (4), 439-462.

Baddeley, A. D. (1990). Human memory: Theory and practice. Boston, USA: Allyn and Bacon.

Bilodeau, E. A., & Howel D. C.. (1965). Free association norms by discrete and continued methods. (Catalog No. D210.2:F87). Washington DC: U.S. Government Print Office.

Bruce, D. & Gaines, M. T. (1976). Tests of an organizational hypothesis of isolation effects in free recall. Journal of Verbal Learning and Verbal Behavior, 15, 59-72.

Budiu, R. & Anderson, J. R. (2004). Interpretation-Based Processing: A unified theory of semantic sentence processing. Cognitive Science, 28, 1-44.

Byrne, M. D. & Anderson, J. R. (1998). Perception and action. In J. R. Anderson & C. Lebiere (Eds.) (1998). The atomic components of thought. Mahwah, NJ: Lawrence Erlbaum Associates, Inc.

Deese, J. (1959). Influence of inter-item associative strength upon immediate free recall. Psychological Reports, 5, 305-312.

Easterbrook, J. A. (1959). The effect of emotion on cue utilization and the organization of behavior. Psychological Review, 66, 183-201.

Gunzelmann, G. & Anderson, J. R. (2003). Problem solving: Increased planning with practice. Cognitive Systems Research, 4, 57-76.

Hirshman, E. (1988). The Expectation Violation Effect - Paradoxical Effects of Semantic Relatedness. Journal of Memory and Language, 27 (1), 40-58.

Hirshman, E., Whelley, M. M. & Palij, M. (1989). An investigation of paradoxical memory effects. Journal of Memory and Language, 28, 594-609.

Hume, D. (1817). A treatise of human nature. London: Thomas and Joseph Allman. In T. H. Leahey (1980). A History of Psychology: Main currents in psychological thought. N.J.: Prentice-Hall.

Hunt, R. R. (1995). The subtlety of distinctiveness: What von Restorff really did. Psychonomic Bulletin & Review, 2(1), 105-112.

Hunt, R. R. & Elliot, J. M. (1980). The role of non-semantic information in memory: Orthographic distinctiveness effects in retention. Journal of Experimental Psychology: General, 109, 49-74. In E. Hirshman (1988). The Expectation Violation Effect - Paradoxical Effects of Semantic Relatedness. Journal of Memory and Language, 27 (1), 40-58.

Hunt, R. R. & Lamb, C. A. (2001). What causes the isolation effect?. Journal of Experimental Psychology: Learning, Memory, & Cognition, 27(6), 1359-1366.

Jarrett, R.F. & Scheibe, K. E. (1963). Association chains and paired-associate learning. Journal of Verbal Learning and Verbal Behavior, 1, 264-268. In D. S. Palermo & J. J Jenkins. (1964b). Word association norms: Grade school through college. Minneapolis: University of Minnesota Press.

Kahneman, D. & Treisman, A. (1984). Changing views of attention and automaticity. In R, Parasuraman & D. R. Davies. Varieties of attention. (1984) Orlando: Academic Press .

Keppel, G. & Strand, B. (1970). Free association responses to the primary purposes and other responses selected from the Palermo-Jenkins norms.. In L. Postman and G. Keppel, Norms of word association (Eds.). New York: Academic Press.

Leahey, T. H. (1980). A History of Psychology: Main currents in psychological thought. N.J.: Prentice-Hall.

Morris, C. D., Bransford, J. D. & Franks, J. J. (1977). Levels of processing versus transfer appropriate processing. Journal of Verbal Learning and Verbal Behavior, 16, 519-533.

Newell, A. (1990). Unified theories of cognition. Cambridge, Mass.: Harvard

University Press. In N. Taatgen (2005a) Architectures of Cognition: Lecture 1. Artificial Intelligence. http://www.ai.rug.nl/avi.

Norman, D. A., & Shallice, T. (1986). Attention to action: Willed and automatic control of behavior. In Baddeley, A. D. (1990a). Human memory: Theory and practice. Boston, USA: Allyn and Bacon.

Palermo, D. S. & Jenkins, J. J. (1964a). Paired-associate learning as a function of the strength of links in the associative chain. Journal of Verbal Learning & Verbal Behavior, 3, 406-412.

Palermo, D. S. & Jenkins, J. J. (1964b). Word association norms: Grade school through college. Minneapolis: University of Minnesota Press.

Pavlik, P. I. & Anderson, J. R. (2003). An ACT-R model of the spacing effect. In F. Detje, D. Doerner, & H. Schaub (Eds.), In Proceedings of the Fifth International Conference on Cognitive Modeling (pp. 177-182). Bamberg, Germany: Universitats-Verlag Bamberg.

Rundus, D. (1971). Analysis of Rehearsal Processes in Free Recall. Journal of Experimental Psychology, 89(1), 63-77.

Schmidt, S. R. (1985). Encoding and retrieval processes in the memory for conceptually distinctive events. Journal of Experimental Psychology: Learning, Memory & Cognition, 11(3), 565-578.

Schmidt, S. R. (1991). Can we have a distinctive theory of memory? Memory & Cognition. 19 , 6, 523-542.

Schunn, C. D. & Wallach, D. (2004). Evaluating goodness-of-fit in comparison of models to data. http://www.lrdc.pitt.edu/schunn/gof/GOF.doc.

Sokolov, E. N. (1963) Perception and the conditioned reflex. New York: Macmillan. In Schmidt, S. R. (1991). Can we have a distinctive theory of memory? Memory & Cognition. 19 , 6, 523-542.

Taatgen, N. (2005a). Architectures of Cognition: Lecture 1. Artificial Intelligence. http://www.ai.rug.nl/avi.

Taatgen, N. (2005b). Architectures of Cognition: Lecture 2. How to make a cognitive model. http://www.ai.rug.nl/avi.

APPENDIX A

SAMPLE EXPERIMENTAL PAIRS LISTS

(USED BY AMSTER ET AL., 1992)

WEAK PAIR LIST		ZERO PAIR LIST	
Cliff	Country	Cliff	Country
Dirt	Earth	Dirt	Earth
Sugar	Sweet	Sugar	Sweet
Blue	Sky	Blue	Sky
Health	Bug	Priest	Bug
Apple	Fruit	Apple	Fruit
Cancer	Disease	Cancer	Disease
Flight	Air	Flight	Air
Face	Dark	Salt	Dark
Floor	Rug	Floor	Rug
Cheese	Mouse	Cheese	Mouse
Sit	Chair	Sit	Chair
Money	Child	Butter	Child
Bed	Sleep	Bed	Sleep
Glass	Window	Glass	Window
Flower	Stem	Flower	Stem
Radio	Mind	Green	Mind
Woman	Man	Woman	Man
Sore	Boil	Sore	Boil

APPENDIX B

EXPERIMENTAL WORD PAIRS

(USED BY AMSTER ET AL., 1992)

	Stimulus	Response	Mean Association Strength	Standard Deviation	Stimulus	Response	Mean Association Strength	Standard Deviation
Filler	Cliff	Country	3.29	1.48	Sore	Boil	3.07	2.09
	Dirt	Earth	5.57	0.73				
Unaccounted Strong Pairs	Sugar	Sweet	5.43	1.12	Cancer	Disease	5.29	0.88
	Woman	Man	5.86	0.35	Bed	Sleep	5.79	0.41
	Glass	Window	5.79	0.56	Cheese	Mouse	5.50	0.91
	Apple	Fruit	5.79	0.41	Floor	Rug	5.50	0.73
Experimental Pairs	Blue	Sky	5.50	0.73	Sit	Chair	5.50	0.73
	Soft	Sky	2.29	1.44	Pretty	Chair	1.93	1.53
	Bread	Sky	0.50	0.73	Anger	Chair	0.29	0.59
	Insect	Bug	4.79	1.37	Baby	Child	5.71	0.45
	Health	Bug	2.93	1.28	Money	Child	2.79	2.37
	Priest	Bug	0.43	1.05	Butter	Child	0.07	0.26
	Flight	Air	5.29	1.10	Flower	Stem	5.21	1.08
	House	Air	2.86	1.73	Tall	Stem	3.57	1.88
	Ruler	Air	0.07	0.26	Queen	Stem	0.21	0.56
	Light	Dark	5.50	1.12	Think	Mind	5.57	0.73
	Face	Dark	2.71	1.94	Radio	Mind	2.14	1.92
	Salt	Dark	0.21	0.56	Green	Mind	0.14	0.52

Filler and Unaccounted Strong Pairs have not been included in the statistical calculations.
Experimental Pairs have been used for the statistical calculations.

APPENDIX C

MODEL LISTING

```
;;; THIS FILE CONTAINS ACT-R 5.0 MODEL FOR EXPECTATION-VIOLATION EFFECT
;;; BY NILUFER OZYORUK-2004
;;;
;;;
;;; Significant parameters:
;;; PARAMETER #1: Association Strengths        (ia)
;;; PARAMETER #2: Retrieval Latency            (:lf)
;;; PARAMETER #3: Effort                       (:effort)
;;; PARAMETER #4: Attention                    (:ga)
;;;
;;; REHEARSAL: pair first, stim, resp later. Rehearse just-encoded pair.
;;; RETRIEVAL calls for pairs first, response words later.
;;; RETRIEVAL blind-alley search cue strategy first, semantic strategy later.
;;;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;  To Run The Model  ;;;;;;;;;;;;;;;;;;;;;;
;;; To run the command line, call
;;; For a single list or single subject run:
;;;   (test1) (test2) (test3) (test4) for the first, second, third and fourth lists
;;; For multiple lists or subjects run:
;;;   (do-experiment 4) can be any number divisible by four
;;; For the whole experiment:
;;;   (do-original) for the original subject #48
;;;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
```

```
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;; Global VARIABLES  ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

(defvar *strong* 0)            ; temporary counter for recalled strong pairs
(defvar *weaker* 0)            ; temporary counter for recalled weaker pairs
(defvar *subj-resp* nil)       ; temporary list of actual responses (per subject)
(defvar *s-list* nil)          ; strong-response-list
(defvar *w-list* nil)          ; weak-response-list
(defvar *z-list* nil)          ; zero-response-list

(defvar *zero-l* nil)     ; list that collects responses from subjects for zero lists
                          ; in a form ((weaker strong))
(defvar *weak-l* nil)     ; list that collects responses from subjects for weak lists
                          ; in a form ((weaker strong))
(defvar *strong-z* 0)            ; # of strongs in zero lists
(defvar *strong-w* 0)            ; # of strongs in weak lists
(defvar *weaker-z* 0)            ; # of weaker in zero lists
(defvar *weaker-w* 0)            ; # of weaker in weak lists
(defvar *total* 0)               ; # of measures
(defvar *res-list* nil)          ; result list

;;;  for effort:
(defvar *effort-coef* 10)        ; effort coefficient
(defvar *remain-effort* 0.0)     ; remaining effort (for blind alley searches)
(defvar *def-prod-time* 0.05)    ; default production (action) time (:dat)
(defvar *ba-loop-time* 0.4)      ; time of an automatic blind-alley search /
                                 ; expectation criteria
```

```
;;;  for re-presentations:
(defvar *add-ref-loop* 1)        ; number of additional references per automatic blind alley search
(defvar *add-ref* 0)             ; number of additional references for blind alley searches

;;; for :ga
(defvar *ga-incr-coef* 1.25)     ; goal activation increase coefficient (1.25 means a 25% increase)


;;;;;;;; Below definition defines the global variable of experimental pairs ;;;;;;;;;;;;;
;;;  with their experimental significances: Filler, Unaccounted Strong, Strong, Weak
;;;  and their associative strengths

(defvar *Amster-exp-data* '(
  ((
    (("cliff" "country")  "F" 3.29)   (("dirt" "earth")   "F" 5.57)     (("sugar" "sweet")  "U" 5.43)
    (("blue" "sky")       "S" 5.50)   (("health" "bug")   "W" 2.93)     (("apple" "fruit")  "U" 5.79)
    (("cancer" "disease") "U" 5.29)   (("flight" "air")   "S" 5.29)     (("face" "dark")    "W" 2.71)
    (("floor" "rug")      "U" 5.50)   (("cheese" "mouse") "U" 5.50)     (("sit" "chair")    "S" 5.50)
    (("money" "child")    "W" 2.79)   (("bed" "sleep")    "U" 5.79)     (("glass" "window") "U" 5.79)
    (("flower" "stem")    "S" 5.21)   (("radio" "mind")   "W" 2.14)     (("woman" "man")    "U" 5.86)
    (("sore" "boil")      "F" 3.07)
   ) "weak")
  ((
    (("cliff" "country")  "F" 3.29)    (("dirt" "earth")   "F" 5.57)    (("sugar" "sweet")  "U" 5.43)
    (("blue" "sky")       "S" 5.50)    (("priest" "bug")   "Z" 0.43)    (("apple" "fruit")  "U" 5.79)
    (("cancer" "disease") "U" 5.29)    (("flight" "air")   "S" 5.29)    (("salt" "dark")    "Z" 0.21)
    (("floor" "rug")      "U" 5.50)    (("cheese" "mouse") "U" 5.50)    (("sit" "chair")    "S" 5.50)
    (("butter" "child")   "Z" 0.07)    (("bed" "sleep")    "U" 5.79)    (("glass" "window") "U" 5.79)
    (("flower" "stem")    "S" 5.21)    (("green" "mind")   "Z" 0.14)    (("woman" "man")    "U" 5.86)
    (("sore" "boil")      "F" 3.07)
   ) "zero")
```

```
((
  (("cliff" "country")  "F" 3.29)   (("dirt" "earth")    "F" 5.57)   (("woman" "man")     "U" 5.86)
  (("bread" "sky")      "Z" 0.50)   (("insect" "bug")    "S" 4.79)   (("glass" "window")  "U" 5.79)
  (("bed" "sleep")      "U" 5.79)   (("ruler" "air")     "Z" 0.07)   (("light" "dark")    "S" 5.50)
  (("cheese" "mouse")   "U" 5.50)   (("floor" "rug")     "U" 5.50)   (("anger" "chair")   "Z" 0.29)
  (("baby" "child")     "S" 5.71)   (("cancer" "disease")"U" 5.29)   (("apple" "fruit")   "U" 5.79)
  (("queen" "stem")     "Z" 0.21)   (("think" "mind")    "S" 5.57)   (("sugar" "sweet")   "U" 5.43)
  (("sore" "boil")      "F" 3.07)
 ) "zero")
((
  (("cliff" "country")  "F" 3.29)   (("dirt" "earth")    "F" 5.57)   (("woman" "man")     "U" 5.86)
  (("soft" "sky")       "W" 2.29)   (("insect" "bug")    "S" 4.79)   (("glass" "window")  "U" 5.79)
  (("bed" "sleep")      "U" 5.79)   (("house" "air")     "W" 2.86)   (("light" "dark")    "S" 5.50)
  (("cheese" "mouse")   "U" 5.50)   (("floor" "rug")     "U" 5.50)   (("pretty" "chair")  "W" 1.93)
  (("baby" "child")     "S" 5.71)   (("cancer" "disease")"U" 5.29)   (("apple" "fruit")   "U" 5.79)
  (("tall" "stem")      "W" 3.57)   (("think" "mind")    "S" 5.57)   (("sugar" "sweet")   "U" 5.43)
  (("sore" "boil")      "F" 3.07)
 ) "weak")
))

(defvar *Amster-data* '(0.28 0.28 0.23 0.43))
```

```
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;;; Below functions shorten the typing for testing the model in a test-by-list manner
;;;;; or the original experiment as the bottom function

(defun test1 ()
  (do-single-list (first *Amster-exp-data*)))

(defun test2 ()
  (do-single-list (second *Amster-exp-data*)))

(defun test3 ()
  (do-single-list (third *Amster-exp-data*)))

(defun test4 ()
  (do-single-list (fourth *Amster-exp-data*)))

(defun do-original ()
  (do-experiment 48))


;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;
;;; This section contains the LISP functions to simulate
;;; the experiment, implement the interface, collect the
;;; data, and display the results
;;;
;;; The ACT-R Model starts further down
;;;
```

```
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;; FOR RUNNING THE EXPERIMENT ;;;;;;;;;;;;;;;;;;;;;;;;;;
;;; This function does the experiment with the equal number of subjects for each list


(defun do-experiment (X)
  (if (zerop (rem X 4))                          ; if the number entered is a multiple of four
      (progn                                     ; then
        (setf *zero-l* nil)                      ; clear zero list responses
        (setf *weak-l* nil)                      ; clear weak list responses
      (setf *total* (* X 2))                     ; set # of measures
        (dotimes (i (/ X 4))                     ; sequentially presents all 4 lists
          (dolist (Y *Amster-exp-data*)          ; (# of subjects divided by 4) times
            (do-single-list Y)
            )
          )
        (analyze-exp-data)                          ; analyze data collected from all subjects
        (correlation *res-list* *Amster-data*)      ; correlate results with real experiment data
        (mean-deviation *res-list* *Amster-data*)   ; mean deviation of results with real experiment data
        (format t "~2%")
        )
    (progn                                          ; else print:
      (format t "~2%There must be equal number of subjects for each list!~%")
      (format t    "Since there are four lists, the number entered must be~%")
      (format t    "a multiple of four.~%")
      )
    )
  )
```

```lisp
(defun do-single-list (L)                          ; single subject experiment; L has to be something
                                                   ; like "first *Amster-exp-data* "
  (setf *subj-resp* nil)                            ; set response list to nil (empty it)
  (setf *def-prod-time* (car (eval `(sgp :dat)))) ; set default production time to global parameter :dat
(default action time) <<<<<<<<<<<<<<
  (present-list (car L))                           ; present list
  (distract)                                       ; 5 minute distractor task
  (present-test)                                   ; present "TEST" and collect data
  (analyze-data L)                                 ; analyze collected data
  (display-results (second L))                     ; display retrieved words
)

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;  PRESENT EXPERIMENTAL LIST ;;;;;;;;;;;;;;;;;;;;;;

(defun present-list (Y)
;--------------------------------------------------------------------------------
; Open a window and display the pair in it

  (let ((window (open-exp-window "Pair List"  :visible t)))

    (reset)
    (pm-install-device window)
    (dolist (x Y)

      (clear-exp-window)
      (add-text-to-exp-window :text (first (car x)) :x 100 :y 125)
      (add-text-to-exp-window :text (second (car x)) :x 150 :y 125)
      (pm-proc-display :clear t)                                    ;attend screen
```

```
;-------------------------------------------------------------------------------
; Calculate remaining effort with respect to the association between stimulus
; and response words by the formula:
;
; = <effort coefficient> * (1 / <association>) * <default action time>
;
; and set the :effort parameter of the "associate" and "rehearse_pair" productions
; through the (set-model-efforts) function, which adds an spp command for the two
; productions below by calculating the effort values based on preset ia's
;;  It is actualized in the "associate" production when the resp is being encoded
;;              also during "rehearse_pair" production during rehearsal
;;  Effort represents the #of blind-alley searches
;;  Latency caused by ia's represent semantic distance

      (setf *remain-effort* (* *effort-coef* (/ 1 (third x)) *def-prod-time*))
      (set-model-efforts)
```

```
;-------------------------------------------------------------------------------
; Create a new chunk of type remember (goal chunks) with the specified slot
; values, add it to the Declarative Memory, and put it into the goal buffer
; (focus on it)

      (let (
            (gname (gentemp "goal"))
            (goal nil)
            )
        (setf goal (list (list gname 'isa      'remember
                                     'purpose "study"
                                     'state   "attending-pair"
                                     'expect  'normal
                                     'context 'experiment)))
        (add-dm-fct goal)
        (goal-focus-fct `(,gname))
        )
;-------------------------------------------------------------------------------
; Set initial goal activation (:ga) for a pair to 1.0 (ACT-R default)

      (sgp-fct '(:ga 1.0))
      (pm-run 10.0 :full-time t)                  ;productions start running after this
      )
    )
  )
```

```
;;;;;;;;;;;;;;;;;;;;;;;;;;;; EXPECTATION-VIOLATION EFFECT FUNCTIONS ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

;;;;;;  The function for allowing RE-PRESENTATIONS to the goal chunks
;;;;;;  The function sets new REFERENCES of a chunk

(defun set-new-ref (chnk)                                           ; set new references of a chunk
  (let (
        (cur-ref (caaar (eval `(sdp ,chnk :references))))          ; retrieve current references of the chunk
        (new-ref nil)
        )
    (setf new-ref (+ cur-ref *add-ref*))                            ; set new references as current references
    (eval `(sdp ,chnk :references ,new-ref))                        ; plus additional references (*add-ref*)
    )
)


;;;;; The function for BLIND-ALLEY SEARCHES
;;;;; The function calculates and sets an :effort of the "associate"  and "rehearse_pair" productions
;;;;; and calculates the additional references with respect to the :effort of the "associate"

(defun set-model-efforts ()                                             ; set efforts in the model
  (if (< *remain-effort* *def-prod-time*)                               ; IF remaining effort is less than :dat
      (progn                                                            ; THEN
        (setf *add-ref* 0)                                              ; set additional references to 0
        (setf *remain-effort* 0)                                        ; set remaining effort to 0
        (eval `(spp associate :effort ,*def-prod-time*))                ; set :effort of the "associate"
production to :dat (0.05) set as safetycheck
        (eval `(spp reherse_pair :effort ,*def-prod-time*))             ; set :effort of the "rehearse_pair"
production to :dat (0.05)
        )
```

```
    (if (> *remain-effort* *ba-loop-time*)                                ; IF remaining effort is greater than
                                                                          ; an blind alley search time
        (progn                                                            ; THEN
          (setf *remain-effort* (- *remain-effort* *ba-loop-time*))  ; set reaming effort current minus an
                                                                          ; automatic blind alley search time
          (setf *add-ref* *add-ref-loop*)                                 ; additional references to default (1)
          (eval `(spp associate :effort ,*ba-loop-time*))                 ; set :effort of the "associate"
                                                                          ; production to blind alley search time
          )
      (progn                                                              ; ELSE
        (eval `(spp associate :effort ,*remain-effort*))                  ; set  :effort of the "associate"
                                                                          ; production to remaining effort
        (eval `(spp rehearse_pair :effort ,*remain-effort*))              ; set :effort of the "rehearse_pair"
                                                                          ; production to remaining-effort
        (setf *remain-effort* 0)                                          ; set remaining effort to 0
        (setf *add-ref* 0)                                                ; set additional references to 0
        )
      )
    )
  )

;;;;; The function for increasing ATTENTION cumulatively
;;;;; The function calculates and sets :ga
;;;;; This function allows the expectation-verification production to do the calculations for criteria
;;;;; (i.e.: establish criteria within expectation-verification production)

(defun increase-ga ()                                     ;increase goal activation (attention)
  (let (
        (cur-ga (car (eval `(sgp :ga))))                  ; retrieve current :ga
        (new-ga nil)
        )
    (setf new-ga (* cur-ga *ga-incr-coef*))               ; set new :ga as current :ga * increase coefficient
    (eval `(sgp :ga ,new-ga))
    )
)
```

```
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;; THE DISTRACTOR TASK ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;  There was a distractor task for 5 minutes between the last pair presented and the start of the
;;;  test session. Distractor tasks are intended to clear the short-term memory, prevent rehearsal
;;;  and allow items which are assumed to be retrieved from the long-term memory.
;;;  This function intends to reproduce the end result of this distractor task by estimating the decay
;;;  that would result from this task.

(defun distract ()
  (let (
        (gname (gentemp "distractor"))
        (goal nil)
        )
    (setf goal (list (list gname 'isa     'remember
                                  'purpose "distractor")))
    (add-dm-fct goal)
    (goal-focus-fct `(,gname))
    (buffers-fct '(=retrieval nil))
    )
  (format t "~2%Distractor Task (5 minutes)~2%")
  (pm-run 300.0 :full-time t)
  )
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;; PRESENT TEST PHASE ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;; presents the word 'test' on the screen
;;; and waits 3 minutes for free-recall responses

(defun present-test ()
```

```
;------------------------------------------------------------------------------
; Open a new window and display word "TEST"

  (let ((window (open-exp-window "Test"  :visible t)))
    (pm-install-device window)
    (clear-exp-window)
    (add-text-to-exp-window :text "test" :x 150 :y 125)
    (pm-proc-display :clear t)
;------------------------------------------------------------------------------
; Create a new chunk with the specified slot values (purpose "test"),
; add it to the Declarative Memory, and put it into the goal buffer
; (focus on it)

    (let (
          (gname (gentemp "test"))
          (goal nil)
          )
      (setf goal (list (list gname 'isa     'remember
                             'purpose "test"
                             'state   "initiating-retrieval"
                                   'context 'experiment)))
      (add-dm-fct goal)
      (goal-focus-fct `(,gname)))
;------------------------------------------------------------------------------
    (format t "~2%Test Session (3 minutes)~2%")
    (pm-run 180.0 :full-time t)                    ;wait 3 minutes for free-recall
    )
  )
```

```
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;; ANALYZE EXPERIMENTAL  DATA ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;  DO STATISTICAL ANALYSIS   ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;        AND  PRINT          ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
(defun analyze-exp-data ()                                         ; analyze experiment data
  (setf *strong-z* 0)                                              ; set all values (counters)
  (setf *strong-w* 0)                                              ; to 0
  (setf *weaker-z* 0)
  (setf *weaker-w* 0)
  (setf *res-list* nil)                                            ; empty result list
  (dolist (z *zero-l*)                                             ; count zero-list results
    (setf *weaker-z* (+ *weaker-z* (first z)))                     ; # of weaker recalls
    (setf *strong-z* (+ *strong-z* (second z)))                    ; # of strong recalls
    )
  (dolist (w *weak-l*)                                             ; count weak-list
    (setf *weaker-w* (+ *weaker-w* (first w)))                     ; # of weaker recalls
    (setf *strong-w* (+ *strong-w* (second w)))                    ; # of strong recalls
    )
  (setf *res-list* (list  (/ (float *strong-w*) (float *total*))  ; make result list by
                          (/ (float *strong-z*) (float *total*))  ; proportions of type of item
                          (/ (float *weaker-w*) (float *total*))  ; in total recall
                          (/ (float *weaker-z*) (float *total*))))
  (format t "~2%")                                                  ; add 2 new lines
  (format t "      Weaker Strong ~%")                               ; write this and insert a new line
  (format t "Zero | ~2,2F | ~2,2F |~%" (/ (float *weaker-z*) (float *total*)) (/ (float *strong-z*) (float *total*)))
  (format t "Weak | ~2,2F | ~2,2F |~2%" (/ (float *weaker-w*) (float *total*)) (/ (float *strong-w*) (float *total*)))
  )
```

```
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;; ANALYZE DATA / PER LIST   ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

(defun analyze-data (L)                                    ; analyze data
  (setf *strong* 0)                                        ; set counter for strong recalls to 0
  (setf *weaker* 0)                                        ; set counter for weaker recalls to 0
  (setf *s-list* nil)
  (setf *z-list* nil)
  (setf *w-list* nil)
  (dolist (x *subj-resp*)                                  ; take list of responses (for each item do:)
    (dolist (y (car L))                                    ; search the presented list for that item
      (if (equal x (second (car y)))                       ; if found than
          (if (equal (second y) "W")                       ; if it is a member of weak pair than
          (progn                                           ; simple grouping function (executes serially)
              (setf *weaker* (+ *weaker* 1))               ; increment *weaker* counter by 1
            (setf *w-list* (push x *w-list*))              ; add word to the weak-response-list
          )
            (if (equal (second y) "Z")                     ; else if it is a member of zero pair than
            (progn
                (setf *weaker* (+ *weaker* 1))             ; increment *weaker* counter by 1
            (setf *z-list* (push x *z-list*))              ; add word to the zero-response-list
            )
              (if (equal (second y) "S")                   ; else if it is a member of strong pair than
              (progn
                  (setf *strong* (+ *strong* 1))           ; increment *strong* counter by 1
              (setf *s-list* (push x *s-list*))            ; add word to the strong-response-list
             )
               nil)))
       nil)))                                                           ; else do nothing
  (if (equal (second L) "zero")                                         ; if the list type is "zero" than
      (setf *zero-l* (push (list *weaker* *strong*) *zero-l*))          ; put the weaker-stronger pair
                                                                        ; into zero-list bin
    (setf *weak-l* (push (list *weaker* *strong*) *weak-l*))))          ; else put the pair into weak-list bin
```

```lisp
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;; PRINT-OUT THE RESPONSE WORDS ;;;;;;;;;;;;;;;;;;;;

(defun display-results (list-type)
  (if (equal list-type "weak")
      (progn
        (format t "~2%")
        (format t "Weak    |")
        (dolist (x *w-list*)                          ; print weak-response-list
          (format t " ~9A " x))
        )
     (progn
       (format t "~2%")
       (format t "Zero    |")
       (dolist (x *z-list*)                           ; print weak-response-list
         (format t " ~9A " x))
       )
     )
  (format t "~%")
  (format t "Strong |")
  (dolist (x *s-list*)                                ; print strong-response-list
    (format t " ~9A " x))
)
```

```
;;;;;;;;;;;;;;;;;;;;;;;; MISCELANEOUS FUNCTIONS ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;; (msp) prints model specific parameters

(defun msp ()
  (format t "~2%Model Specific Parameters:~%")
  (format t    "-------------------------------------~%")
  (format t     "effort coefficient            : ~5,2F~%" *effort-coef*)
  (format t     "auto BA search time           : ~5,2F~%" *ba-loop-time*)
  (format t     "references per auto BA search : ~5,2F~%" *add-ref-loop*)
  (format t     "attention increase coefficient: ~5,2F~2%" *ga-incr-coef*)
  )
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

(clear-all)
(pm-reset)

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;; PARAMETERS   ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
(sgp :v t :esc t    :bll .5  :lf 1 :rt 1.15 :lt t :ans .1 )
(pm-set-params :real-time nil :show-focus t )

;;; bll to bring decay rate into the equation
;;; rt  to determine retrieval threshold for chunks   :rt 1.15
;;; lf  to increase the rehearsal & retrieval time    :lf 1
;;; ans (activation noise s)                          :ans .1
```

```
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;; CHUNK-TYPES   ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

(chunk-type meaning                  ;represents the word
        word                         ;the string which presents the word
      context                        ;the pair (as id#) that the words belongs to
      role                           ;whether the word is a stimulus or a response
)

(chunk-type pair                     ;represents the pair
      expect                         ;normal or surprise (initially normal)
                                     ;Ss' expectation that the list will have a certain property
      context                        ;will be experiment, in order to spread activation thru the list
)

(chunk-type remember                 ;represents the task context
        stim                         ;stim is word1
        resp                         ;resp is word2
      pair                           ;indicates the semantic pair id#
      expect                         ;expectation carries 'normal' or 'surprise' values
      context                        ;experiment
        purpose                      ;study or test purpose
      state
)
```

```
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;   CHUNK DEFINITIONS   ;;;;;;;;;;;;;;;;;;;;;;;;;;;;
(add-dm
      (cliff* isa meaning word "cliff")
      (country* isa meaning word "country")
      (dirt* isa meaning word "dirt")
      (earth* isa meaning word "earth")
      (sore* isa meaning word "sore")
      (boil* isa meaning word "boil")
      (sugar* isa meaning word "sugar")
      (sweet* isa meaning word "sweet")
      (woman* isa meaning word "woman")
      (man* isa meaning word "man")
      (glass* isa meaning word "glass")
      (window* isa meaning word "window")
      (apple* isa meaning word "apple")
      (fruit* isa meaning word "fruit")
      (cancer* isa meaning word "cancer")
      (disease* isa meaning word "disease")
      (bed* isa meaning word "bed")
      (sleep* isa meaning word "sleep")
      (cheese* isa meaning word "cheese")
      (mouse* isa meaning word "mouse")
      (floor* isa meaning word "floor")
      (rug* isa meaning word "rug")
      (sky* isa meaning word "sky")
      (blue* isa meaning word "blue")
      (soft* isa meaning word "soft")
      (bread* isa meaning word "bread")
      (bug* isa meaning word "bug")
      (insect* isa meaning word "insect")
      (health* isa meaning word "health")
```

```
      (priest* isa meaning word "priest")
      (air* isa meaning word "air")
      (flight* isa meaning word "flight")
      (house* isa meaning word "house")
      (ruler* isa meaning word "ruler")
      (dark* isa meaning word "dark")
      (light* isa meaning word "light")
      (face* isa meaning word "face")
      (salt* isa meaning word "salt")
      (chair* isa meaning word "chair")
      (sit* isa meaning word "sit")
      (pretty* isa meaning word "pretty")
      (anger* isa meaning word "anger")
      (child* isa meaning word "child")
      (baby* isa meaning word "baby")
      (money* isa meaning word "money")
      (butter* isa meaning word "butter")
      (stem* isa meaning word "stem")
      (flower* isa meaning word "flower")
      (tall* isa meaning word "tall")
      (queen* isa meaning word "queen")
      (mind* isa meaning word "mind")
      (think* isa meaning word "think")
      (radio* isa meaning word "radio")
      (green* isa meaning word "green")
      (test* isa meaning word "test")
)

;;;  Added sdp (set declarative parameters) command in order to set creation time and references
;;;  to show the words have been created a long time ago and have been used many times.
;;;  This increases the base-level activation so no need to use :blc
;;   Base-levels increase from for eg: .069 with 100 references to 2.303 with 500 references

(sdp :references 500 :creation-time -10000)
```

```
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;  PRODUCTIONS  ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;


;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;; STUDY SESSION ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
(p attend-pair
      =goal>
            isa         remember
            context     =context                ;defined initially in the functions (experiment)
                                                ;context defined as 'experiment'
                                                ;provides activation to spread
            stim        nil
            resp        nil
            expect      =expectation            ;defined initially in the functions (normal)
            state       "attending-pair"
      =visual-location>
            isa         visual-location
      =visual-state>
            isa         module-state
            modality    free
==>
      =pair>                                    ;acknowledges a pair seen on the screen
            isa         pair                    ;allows for a pair chunk to be created,
                                                ;and notes its random chunk id#
            expect      =expectation            ;assigns the initial expectation (normal) to the seen pair
            context     =context                ;acknowledges this pair is presented during the experiment
      +visual>
            isa         visual-object
            screen-pos  =visual-location
      =goal>
            isa         remember
            state       "reading"
            pair        =pair                   ;gives the pair# to the pair slot for identification
            context     =context                ;experiment
)
```

```
(p read-the-word
      =goal>
            isa          remember
            state        "reading"
      =visual>
            isa          text
            value        =val
==>
      =goal>
            isa          remember
            state        "encode-meaning"
      +retrieval>
            isa          meaning
            word         =val
)

(p encode-meaning-stim-and-find-resp
      =goal>
            isa          remember
            stim         nil
            resp         nil
            pair         =pair                ;will indicate the stimulus belongs to the appropriate pair
            expect       =expectation         ;is already determined to be normal
            state        "encode-meaning"
      =retrieval>                             ;retrieves the studied word's meaning
            isa          meaning
            word         =stim                ;states the string the meaning is attached to
```

```
==>
      =retrieval>                              ;encodes the studied word's meaning
            isa         meaning
            word        =stim                  ;states the string the meaning is attached to
            context     =pair                  ;indicates the background context is a pair (with id#)
            role        stim                   ;indicates the studied word is a stimulus
        +visual-location>
            isa         visual-location
            screen-x    greater-than-current
            nearest     current
      =goal>
            isa         remember
            stim        =retrieval             ;encodes retrieved meaning as the stimulus part of the pair
            pair        =pair                  ;encodes the appropriate pair (with id#)
            state       "finding-response"
)

(p attend-response
      =goal>
            isa         remember
            state       "finding-response"
      =visual-location>
            isa         visual-location
      =visual-state>
            isa         module-state
            modality    free
==>
      +visual>
            isa         visual-object
            screen-pos  =visual-location
      =goal>
            isa         remember
            state       "reading"
)
```

```
(p encode-meaning-response
      =goal>
            isa         remember
            stim        =stim
            resp        nil
            pair        =pair
            expect      =expectation
            state       "encode-meaning"
      =retrieval>
            isa         meaning
            word        =resp
==>
      =retrieval>
            isa         meaning
            word        =resp                ;encodes the studied word's meaning
            context     =pair                ;indicates the background context is a pair (with id#)
            role        resp                 ;response word
      =goal>
            isa         remember
            stim        =stim
            resp        =retrieval           ;encodes retrieved meaning as the response part of the pair
            pair        =pair                ;encodes the appropriate pair (with id#)
            purpose     "study"
            state       "re-attending-pair"
      -visual-location>
)
```

```
(p associate                              ;;; this production is a theoretical extension of
      =goal>                              ;of re-attending to the pair as an evaluation of
            isa         remember          ;semantic association between stimulus & response
            stim        =stim             ;effort level is based on ia
            resp        =resp             ;(effort is set within the present-list function in the
beginning)
            state       "re-attending-pair"     ;and applied to this production
==>
      =goal>
            isa         remember
            state       "rehearsing"
      !eval!  (set-new-ref (goal-focus))  ; add re-presentations (:references to goal chunk)
)

(p expectation-verification               ;;; this production will decide based on a criteria
      =goal>                              ; whether the pair has established association yet
            isa         remember          ; if not, grant an extra attention for the task context
            state       "rehearsing"      ; and order pair to go to associate prod to be re-presented
            expect      normal            ;this is only theoretical, implementing the conflict
      !eval! (> *remain-effort* 0)        ;;; if *remain-effort* in not 0
                                          ; (theoretically: if there is a failure to associate)
==>
      =goal>
            isa         remember
            state       "re-attending-pair"     ; continue an association attempt
      !eval!  (increase-ga)                     ; increases the attentional level on the pair
      !eval!  (set-model-efforts)               ; re-calculates the effort level of the pair
                                                ; (computationally necessary)
)                                               ; theoretically not done by expectation-verification,
                                                ; but computationally the pair's effort level needs
                                                ; to be calculated before it enters 'associate' again
```

```
(p review                                              ; sets the 'surprise'tag
      =goal>                                           ; based on attention levels
            isa          remember
            pair         =pair
            state        "rehearsing"
      =pair>
            isa          pair
          -   expect     surprise                      ;if the pair is not tagged with 'surprise'
      !eval! (zerop *remain-effort*)                   ;if remaining effort is 0
      !eval! (> (car (sgp :ga)) 1)                     ;if :ga is greater than 1 (criterion)
==>
      =pair>
            isa          pair
            expect       surprise                      ; tag the pair as 'surprising'
      =goal>
            isa          remember
            expect       surprise                      ;change the expectation of the pair as surprised
            state        "rehearsing"
)

;;;;;;;;;;;;;;;;;;;;;;;;  REHEARSAL  ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;
;;; Below productions show a two-stage REHEARSAL process where initially the pair concept is rehearsed
;;; and then the stimulus & response word items are rehearsed individually too
;;; Rehearsal borrowing is not allowed
```

```
(p initiating_rehearsal                           ;initiation of the rehearsal loop
      =goal>
            isa          remember
            pair         =pair
            state        "rehearsing"
==>
      +retrieval>        =pair                    ;request retrieval for the pair
                                                  ;with the appropriate pair id# for the first rehearsal
      =goal>
            isa          remember
            state        "rehearsing_pair"
)

                                                  ;;; Rehearsing pair does not retrieve words thus it is
(p rehearse_pair                                  ;;  simply the 'concept' of the pair and can be used
      =goal>                                      ;;  as a cue
            isa          remember
            state        "rehearsing_pair"
      =retrieval>                                 ;retrieval adds :references to the pair
            isa          pair
            context      =experiment
==>
      +retrieval>
            isa          meaning                  ;request retrieval for word items
            context      =retrieval               ;which belong to context of same pair
            role         stim
      =goal>
            isa          remember
            state        "rehearsing_stim"
)
```

```
(p rehearse_stimulus
      =goal>
            isa          remember
            state        "rehearsing_stim"
      =retrieval>
            isa          meaning
            word         =stim
            context      =pair
            role         stim
==>
      +retrieval>                            ;request retrieval for word items
            isa          meaning
            context      =pair               ;request specific pair
            role         resp                ;retrieve the response
      =goal>
            isa          remember
            state        "rehearsing_resp"
)
(spp rehearse_stimulus :effort 0.35)         ;sets the effort level for rehearsing stimulus word

(p rehearse_response
      =goal>
            isa          remember
            state        "rehearsing_resp"
            stim         =stim
            context      =experiment
      =retrieval>
            isa          meaning
            word         =resp
            context      =pair
            role         resp
==>
      +retrieval>        =pair               ;retrieval request for the same pair
      =goal>                                 ; (because rehearsal borrowing is not allowed)
            isa          remember
            state        "rehearsing_pair"
)
(spp rehearse_response :effort 0.35)         ;sets the effort level for rehearsing response word
```

```
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;; TEST SESSION ;;;;;;;;;;;;;;;;;;;;;;;
;;; Recall session starts with clear retrieval and goal buffers carrying no pair information.
;;; The recall starts with surprise cue strategy, later continues with semantic strategy.
;;; The functions automatically switch goal purpose to "test" so the productions will not
;;; be mixed up.
;;; and the state to "initiating-retrieval" as soon as the word "test" is presented

(p retrieve-instance
        =goal>
                isa             remember
                context         =experiment
                state           "initiating-retrieval"
                purpose         "test"
==>
        +retrieval>
                isa             remember            ;requests retrieval for study instance
                purpose         "study"
        =goal>
                isa             remember
                state           "retrieving-instance"
)

(p retrieving-from-instance
        =goal>
                isa             remember
                context         =experiment
                state           "retrieving-instance"
                purpose         "test"
        =retrieval>
                isa             remember
                expect          =expect
                stim            =stim
                pair            =pair
                resp            =resp
```

```
==>
      =pair>                                    ;to make sure that the pair will not be attempted
            isa         pair                    ;to be retrieved
            context     "retrieved"
        =retrieval>
            isa         remember
            purpose     "retrieved"             ; tag the instance as retrieved
        =goal>
            isa         remember
            pair        =pair
            expect      =expect                 ;place expectation here so activation will spread to others
            state       "reporting-from-inst"
        +retrieval>     =resp                   ;requests retrieval of the response word
)

(p reporting-from-instance
      =goal>
            isa         remember
            state       "reporting-from-inst"
            purpose     "test"
      =retrieval>
            isa         meaning
            word        =val
==>
      =retrieval>
            isa         meaning                 ;;; tags the retrieved word as such
            role        "retrieved"             ;;  to prevent further attempts for retrieval
      =goal>
            isa         remember
            state       "initiating-retrieval"
      !eval!  (setf *subj-resp* (push =val *subj-resp*))    ;;; Adds the retrieved
                                                             ;; word into a list named
                                                             ;; "subj-resp" to be reported
                                                             ;; at the end of the test
                                                             ;; session
)
```

```
(p change-strategy
        =goal>
                isa             remember
                context         =experiment
                state           "retrieving-instance"
                purpose         "test"
        =retrieval>
                isa             error
==>
        =goal>
                isa             remember
                expect          nil
                state           "initiate-semantic-retrieval"
)

(p retrieve-pair
        =goal>
                isa             remember
                context         =experiment
                state           "initiate-semantic-retrieval"
                purpose         "test"
==>
        +retrieval>
                isa             pair                ;requests retrieval  for semantic pair
                context         =experiment
                                                    ;;; no request for any expectation type,
        =goal>
                isa             remember
                purpose         "test"
                state           "recalling-pair"
)
```

```
(p retrieving-from-pair
        =goal>
            isa            remember
            state          "recalling-pair"
            purpose        "test"
      =retrieval>
            isa            pair
==>
      =retrieval>
            isa            pair
            context        "retrieved"        ;;; tags the retrieved pair as such
        +retrieval>
            isa            meaning            ;requests retrieval for response word
            context        =retrieval
            role           resp
        =goal>
            isa            remember
            purpose        "test"
            pair           =retrieval
            state          "reporting"
)

(p reporting-from-pair
      =goal>
            isa            remember
            state          "reporting"
            purpose        "test"
      =retrieval>
            isa            meaning
            word           =val
```

```
==>
      =retrieval>
            isa         meaning           ;;; tags the retrieved word as such
            role        "retrieved"       ;;  to prevent further attempts for retrieval
      =goal>
            isa         remember
            state       "initiate-semantic-retrieval"
      !eval!   (setf *subj-resp* (push =val *subj-resp*))   ;;; Adds the retrieved
                                                              ;; word into a list named
                                                              ;; "subj-resp" to be reported
                                                              ;; at the end of the test
                                                              ;; session
)

(p end-recall
        =goal>
            isa         remember
            state       "recalling-pair"              ;;; This assumes the 'pair' retrieval will be
            purpose     "test"                        ;    the reason of retrieval failure and
        =retrieval>                                   ;    not the failure of 'word' retrieval.
            isa         error
 ==>
      =goal>
            isa         remember
            pair        nil
            state       "stop"
)
```

```
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;; COMMANDS ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
(setf *actr-enabled-p* t)
(pm-set-visloc-default :attended new :screen-x lowest)     ;shows where to attend in the new screen

;;;;;;;;;;
;;; Below are the original association strengths from the Amster & Ozyoruk (1992) norms
;;; They are based on 7-point scale
;;; Strongly associated pairs have association strengths between 6 and 5
;;; Weakly associated pairs have association strengths between 3.5 and 2
;;; Zero association pairs have association strengths between 0.5 and 0

;;; This list contains the original association strengths -2 in order to achieve negative priming

;;; Original 7-point scale (-2)
 (Set-IA (cliff* country* 1.29)     (blue* sky* 3.50)        (priest* bug* -1.57)
      (dirt* earth* 3.57)           (health* bug* 0.93)      (house* air* 0.86)
      (sore* boil* 1.07)            (flight* air* 3.29)      (ruler* air* -1.93)
      (sugar* sweet* 3.43)          (face* dark* 0.71)       (light* dark* 3.50)
      (woman* man* 3.86)            (sit* chair* 3.50)       (salt* dark* -1.79)
      (glass* window* 3.79)         (money* child* 0.79)     (baby* child* 3.71)
      (apple* fruit* 3.79)          (flower* stem* 3.21)     (butter* child* -1.93)
      (cancer* disease* 3.29)       (radio* mind* 0.14)      (pretty* chair* -0.07)
      (bed* sleep* 3.79)            (soft* sky* 0.29)        (anger* chair* -1.71)
      (cheese* mouse* 3.50)         (bread* sky* -1.5)       (tall* stem* 1.57)
      (floor* rug* 3.50)            (insect* bug* 2.79)      (queen* stem* -1.79)
      (think* mind* 3.57)           (green* mind* -1.86)
)
```