Explorations in ACT-R Based Cognitive Modeling – Activation, Selection and Verification without Inhibition in Language Analysis

Jerry T. Ball

Air Force Research Laboratory
Wright-Patterson Air Force Base
Jerry.Ball1@wpafb.af.mil

Abstract

This paper explores the benefits and challenges of using the ACT-R cognitive architecture in the development of a large-scale, functional, cognitively motivated language analysis model. The paper focuses on ACT-R's spreading activation mechanism, proposing extensions to support multi-level activation spread and carry over activation. The paper argues against the need for inhibition which is necessarily task specific, proposing instead, the use of ACT-R's general declarative memory (DM) retrieval (i.e., activation and selection) mechanism, supplemented with a verification mechanism, combined with task specific productions and context biasing to support language analysis.

Introduction

Our research team has been working on the development of a language analysis model (Ball, 2011; Ball, Heiberg & Silber, 2007) within the ACT-R cognitive architecture (Anderson, 2007) since 2002 (Ball, 2004). The focus is on development of a general-purpose, large-scale, functional model (Ball, 2008; Ball et al., 2010) that adheres to well established cognitive constraints on human language processing (HLP) as realized by ACT-R. Two important cognitive constraints on HLP that we adhere to are *incremental* and *interactive* processing (Just & Carpenter, 1987; Altmann & Steedman, 1988; Tanenhaus et al., 1995; Altmann, 1998; Gibson & Pearlmutter, 1998).

ACT-R incorporates two architectural constraints, realized as serial bottlenecks, which largely determine incremental processing: 1) a single production can execute at a time, and 2) a single declarative memory (DM) chunk can be retrieved at a time. Outside these serial constraints which are the basis of *incremental* processing, ACT-R provides architectural support for parallel processing in the form of a parallel production selection mechanism based on utility and a parallel DM spreading activation mechanism. These parallel mechanisms are probabilistic and context dependent. The parallel/probabilistic mechanisms provide the basis for *interactive* processing. They guide the behavior of the language analysis model in directions that are likely to lead to a successful analysis given the current context and current input. The basic processing cycle in ACT-R—called the *procedural loop*—involves the selection and execution of a sequence of productions. Production execution can result in a perceptual-motor action (e.g. visual attention shift, mouse movement), a modification to the contents of a buffer, or a DM retrieval. When the executing production invokes a DM retrieval, the parallel spread of activation to chunks in DM (*soft constraints* or biases) is followed by retrieval of the single most highly activated chunk which matches a retrieval template (*hard constraint*) specified by the executing production. Activation spreads to DM from the contents of buffers which are the sources of activation (i.e. the current context). The combination of serial and parallel processing gives ACT-R a focused, serial, incremental processing capability which operates over a diffuse, parallel, and highly interactive, context sensitive substrate.

The language analysis model makes extensive use of ACT-R's serial and parallel processing mechanisms. The model processes the linguistic input incrementally, one word or multi-word unit at a time, and uses all available information interactively to make the best choice at each choice point. The model also relies on a non-monotonic mechanism of *context accommodation* which is capable of making modest adjustments to the evolving representation when the current input, in combination with the current context, indicates the need for such accommodation. Context accommodation is part of normal processing—in the right context, a production capable of accommodating the input executes. For example, in incrementally processing "the airspeed restriction", when "airspeed" is processed, it is integrated as the head of the nominal (i.e. noun phrase) projected during the processing of "the", but when "restriction" is subsequently processed, the model accommodates "restriction" by shifting "airspeed" into a modifier function and making "restriction" the head of the nominal. Context accommodation is not capable of handling the kinds of disruptive garden path sentences that are a mainstay of psycholinguistic research (e.g. Bever's (1970) famous "the horse raced past the barn fell"). Such inputs require reanalysis mechanisms which have not yet been implemented. The focus of model development is on handling common or garden English—inputs which humans process with ease, but which, nonetheless, present significant modeling challenges. The combination of parallel/probabilistic processing, and serial processing with context accommodation allows the model to pursue the single best analysis, but to adjust the analysis without

backtracking or reanalysis, when needed. The overall effect is a *pseudo-deterministic* HLP which presents the appearance and efficiency of deterministic processing, despite the rampant ambiguity which makes truly deterministic processing impossible (Ball, 2011).

The output of language analysis is a collection of integrated DM chunks that capture the grammatical structure, and to some extent, the meaning of the input. The overall structure reflects the integration of grammatical constructions projected from lexical items in the input (Ball, 2007b). For debugging and demo purposes, we have created a capability to generate diagrammatic trees from the integrated chunks (Heiberg, Harris & Ball, 2007). Shown below is the diagrammatic tree that is output for the input "what could he have been given":

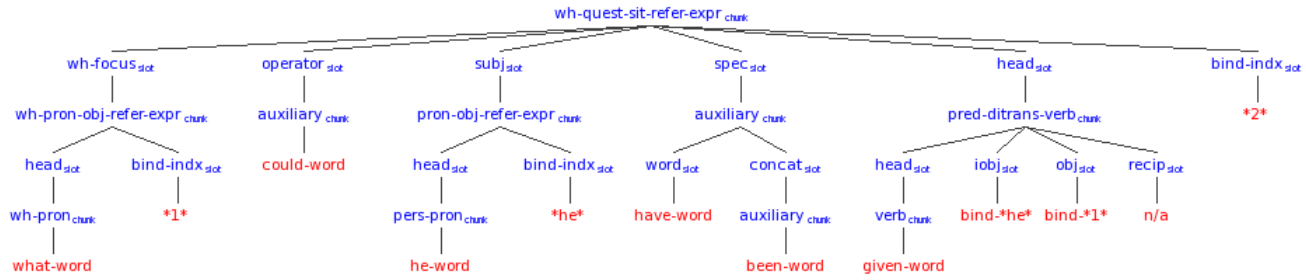

Figure 1: Diagrammatic tree for "What could he have been given?"

This output demonstrates many of the capabilities of the language analysis model. It shows the integration of five distinct constructions and demonstrates many of the representational features of the grammar:

The term *situation referring expression* (sit-refer-expr) corresponds to clause or sentence in other formalisms. The term *object referring expression* (obj-refer-expr) corresponds to a nominal or noun phrase in other formalisms. The use of these terms derives from the focus on encoding referential meaning (i.e. situation referring expression and object referring expression are the two main types of referring expressions) and from the claim that the representations are not purely syntactic (Ball, 2007a).

Wh-question construction (wh-quest-sit-refer-expr): Projected during the processing of the auxiliary verb "could" in the context of the wh-word "what". Projection of the wh-question construction leads to integration of "what" as the wh-focus and "could" as the operator, and predicts the occurrence of the grammatical functions subject (subj), specifier (spec) and head. These predicted functions may be filled by a range of different linguistic forms (e.g. the subject function may be filled by a pronoun as in the example, a full nominal as in "what could *the man* have been given" or even a clausal form as in "what could *going to the movies* have to do with making money"). These linguistic forms may themselves specify further grammatical functions. For example, the ditransitive verb construction which functions as the clausal head predicts an indirect object (iobj), object (obj) and recipient (recip). Note that "what" is additionally bound to the object function of the ditransitive verb construction via the value bind-*1* which matches the bind-indx of "what" (*1*). The binding of "what" to the object (and not the indirect object) of "given" is based on the inanimacy of "what" which is typical of objects and atypical of indirect objects (the animacy feature which drives this decision is not shown in this diagram). The binding of "what" to the object function occurs during the processing of "given".

Ditransitive verb construction (pred-ditrans-verb). Projected from "given" and integrated as the head of the wh-question, with predictions for an object, indirect object and recipient.

Nominal constructions (wh-pron-obj-refer-expr; pron-obj-refer-expr). Projected from "what" (wh-pronoun object referring expression) and "he" (pronoun object referring expression).

Auxiliary construction. Projected from the auxiliary verb "have" with a prediction for a second auxiliary (via the concat slot) which is filled by "been". The auxiliary "been" predicts a third auxiliary, which is not fulfilled, via its own concat slot. Unfilled concat slots are not displayed.

Passive construction. Indicated by the verb form "given" (passive participle form), the passive voice feature (the voice feature is not shown in this diagram), and the binding of the subject "he" to the indirect object via bind-*he*. This binding is based on the human animacy feature of "he", and the passive construction. The binding occurs during the processing of "given". If the input had been "who could it have been given", the object and indirect object bindings would have been reversed (i.e. who = indirect object, it = object).

The diagram is a simplification of the integrated construction chunks. It does not show all the slot values of all the chunks. For example, in this diagrammatic version, it does not show grammatical features like tense, aspect, voice, animacy, number, gender etc., although these features can optionally be shown.

To demonstrate incremental processing and show grammatical features, the next two diagrams show the result of processing "he" followed by "gave" as in "he gave me it":

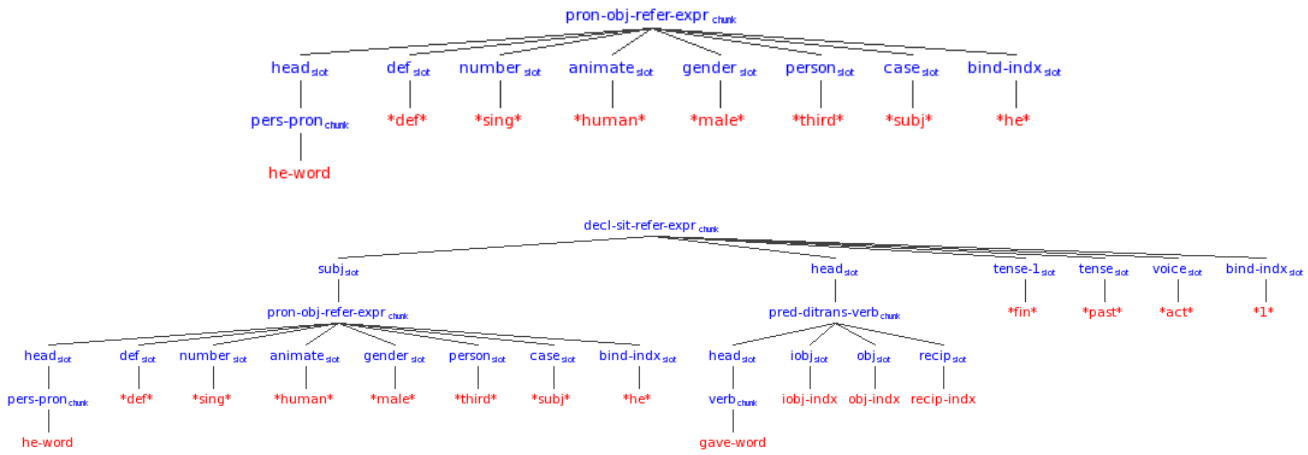

Figure 2: Incremental diagrammatic trees for "he" followed by "gave"

The incremental processing of the (personal) pronoun "he" (pers-pron) leads to projection of a pronoun object referring expression (pron-obj-refer-expr) or nominal. As this diagram shows, pronouns like "he" encode several grammatical features including definiteness (def *def*), number (number *sing*), animacy (animate *human*), gender (gender *male*), person (person *third*) and case (case *subj*).

The processing of the verb "gave" following "he" leads to projection of a declarative situation referring expression (decl-sit-refer-expr) or clause ("he" alone does not indicate a situation). The nominal that was projected from "he" is integrated as the subject and the predicate ditransitive verb (pred-ditrans-verb) construction projected from "gave" is integrated as the head. Note the incremental prediction for the occurrence of an indirect object (iobj iobj-indx), object (obj obj-indx) and recipient (recip recip-indx) following the ditransitive verb "gave" (e.g. "he gave me the ball" or "he gave the ball to me"). The model allows either an indirect object (e.g. "me") or recipient (e.g. "to me") to be expressed following "gave", but not both. This constraint is enforced by the productions which execute following "gave" during the processing of the subsequent input. If the input terminated with "gave", the predictions for an indirect object, object and recipient would still hold and the model might look for them elsewhere (as was the case in the example "what could he have been given"). The diagram also shows the grammatical features finite, past tense (tense-1 *fin*; tense *past*) and active voice (voice *act*) which were projected from the verb to the clause.

To support the creation of integrated linguistic representations, the language analysis model uses a collection of buffers to retain the partial products of language analysis. These buffers are functionally needed to support language analysis. We view the collection of buffers as the ACT-R/language analysis model equivalent of Baddeley's Episodic Buffer (Baddeley, 2000). According to Baddeley, "The episodic buffer is assumed to be a limited-capacity temporary storage system that is capable of integrating information from a variety of sources…the buffer provides not only a mechanism for modeling the environment, but also for creating new cognitive representations" (ibid, p. 421). A key empirical result which motivated Baddeley to introduce the episodic buffer after 25 years of working memory research is the *prose recall* effect. Subjects are capable of recalling greater than 16 words (without error) in the context of a text passage, whereas they are limited to recalling about 5 words (without error) in isolation. This prose recall capability greatly exceeds the capacity of the phonological loop and appears to be based on a chunking (or integration) mechanism. Evidence of a prose recall effect in patients with a severely degraded long-term memory (LTM) capacity argues against an LTM explanation (LTM ~ DM in the ACT-R architecture). Ultimately, the empirical evidence led Baddeley to propose the addition of the episodic buffer to his model of working memory.

Currently, the language analysis model comprises more than 63,000 declarative memory (DM) elements (including more than 61,000 lexical items) and ~700 (grammatical) productions, and is capable of processing a broad range of English language constructions (Ball, Heiberg & Silber, 2007; http://doublertheory.com/comp-grammer/comp-grammar.htm). Our ultimate goal is to be able to handle unrestricted text at levels comparable to leading computational linguistic systems (refs) without extensive training on any specific corpus—as is true of humans, but not computational linguistic systems. In terms of processing speed, the model is capable of processing ~300 written words per minute (wpm) on a quad-core, 64-bit machine running Windows 7 and Allegro Common Lisp (ACL) (64-bit software versions). In addition, ACT-R supports the measurement of cognitive processing time, and the model is capable of processing ~140 wpm in ACT-R cognitive processing time. This compares to a cognitive processing range of between 200 and 300 wpm for adult readers of English (for full comprehension). To bring the model into closer alignment with adult reading rates, we are working on reducing both the number of productions which must execute, and the number of DM elements which must be retrieved, during language analysis (Freiman & Ball, 2010).

Activation

In ACT-R, all DM chunks have an activation level which depends on the current context and prior history of use of the chunk. The prior history of use is expressed via the base level activation, and the current context is reflected in the spreading activation mechanism. A key assumption is that the current context is captured in the contents of the ACT-R buffers which are sources of activation. The most basic form of the activation equation is shown below where A_i = total activation of chunk i; B_i = base level activation of chunk i, and S_i = spreading activation contribution to activation of chunk i (activation equations and descriptions are taken from the ACT-R 6.0 Reference Manual).

$$A_i = B_i + S_i$$

The base level activation is a logarithmic function of the number of uses of a chunk over time combined with a negative exponential decay mechanism. The basic equation is shown below where n = the number of uses of chunk i; t_j = the time since the jth presentation, and d = the decay parameter:

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

This equation is approximated by a computationally simpler "optimized learning" equation (which we are using in our modeling efforts) where n = the number of presentations of chunk i; L = the lifetime of chunk i (the time since its creation), and d = the decay parameter. In this simpler equation, each use is not time dependent.

$$B_i = \ln(n/(1-d)) - d * \ln(L) + \beta_i$$

Spreading activation to a chunk is computed using following equation:

$$S_i = \sum_k \sum_j W_{kj} S_{ji}$$

The elements k are summed over all of the chunks in buffers in the model. The elements j are summed over the chunks which are in the slots of the chunk in buffer k (these are referred to as the sources of activation). W_{kj} = the amount of activation from source j in buffer k. It is the source activation of buffer k divided by the number of sources j in that buffer (by default). S_{ji} = the strength of association from source j to chunk i. The strength of association, S_{ji}, between two chunks is 0 if chunks j and i are not the same chunk and j is not in a slot of chunk i. If chunks j and i are the same chunk or chunk j is in a slot of chunk i the S_{ji} is set based on the fan equation:

$$S_{ji} = S - \ln(fan_{ji})$$

Where S = the maximum associative strength. fan_{ji} is typically thought of as the number of chunks in which j is the value of a slot plus one for chunk j being associated with itself. Note that the fan effect reflects the prior history of use of a chunk even though it is part of the spreading activation equation and not the base level activation.

The activation of eligible DM chunks is computed at the time a DM retrieval request is processed during the execution of a production which attempts a retrieval. A retrieval template is provided on the right hand side of the production attempting a retrieval. The retrieval template determines which DM chunks are eligible to be retrieved. The retrieval template imposes *hard constraints* on retrieval (i.e. an exact match is required unless partial matching, which we do not use, is turned on). The spreading activation mechanism imposes *soft constraints* or biases on retrieval (i.e. the context does not have to match for a chunk to be retrieved).

We make extensive use of ACT-R's activation and retrieval mechanism in the language analysis model. In the word recognition subcomponent, a perceptual span which encodes the visual contents of the current attention fixation spreads activation to DM and the word or multi-word unit which is most highly activated is retrieved and compared to the perceptual input. If the comparison is close enough, the retrieved word or multi-word unit is considered a match. Overall, the process involves four steps: 1) perceptual encoding of the input (*encoding*); 2) activation of declarative memory (*activation*); 3) retrieval of the most highly activated DM chunk which matches the hard constraints of the retrieval template (*selection*); and 4) comparison of the retrieved memory element against the perceptual input (*verification*). Completion of all but the third step presents challenges for ACT-R based modeling.

ACT-R's built in perceptual encoding mechanism assumes words are divided into units by spaces and automatically separates punctuation into separate perceptual units. While this typically succeeds in identifying words and punctuation, it often does not. There are words like "etc." and "didn't" which incorporate punctuation and there are words like "a priori" and "none the less" which have spaces. In addition, the model includes multi-word units like "get out" and "New York" which are encoded in the mental lexicon as lexical items. Higher level knowledge from the mental lexicon is needed to decide what constitutes a word or multi-word unit. To support the integration of higher level knowledge with perceptual processing, we modified ACT-R's perceptual encoding mechanism to incorporate a perceptual span that does not automatically segment the input at spaces and punctuation (Freiman & Ball, 2010). Not only does the perceptual span mechanism support the integration of higher level knowledge, it speeds up processing significantly since words like "don't" are recognized as a single unit instead of three separate units "don", "'" and "t". Likewise, multi-word units like "get out" are also recognized as a unit. Besides speeding up processing, the recognition of multi-word units reduces ambiguity significantly. The word "take" is extremely ambiguous, whereas multi-word units like "take out", "take off" and "take in" are far less ambiguous. The ability to recognize multi-word expressions is an important tool for

handling the ambiguity of natural language and for speeding up the model. We view the addition of multi-word expressions as the best way of achieving adult reading rates.

With a mental lexicon larger than 61,000 lexical items (what we call a *comprehensive mental lexicon*), the computation of activation presents a serious computational challenge. It is not possible to compute the activations of 61,000 plus lexical items prior to each retrieval, in real-time, on existing hardware. As a workaround, we developed a capability to minimize the activation computations in the event of an exact match to the form of the input. If there is a lexical item in DM which is an exact match to the perceptual span, a hard constraint is added to the retrieval template to restrict the number of matching DM elements. When the full perceptual span doesn't match, the match is backed off to the last space in the perceptual span and re-attempted. Prior spaces can also be backed off to. If there is no match (e.g. if the input is "spped")—as a computational compromise—the model attempts a retrieval requiring a hard constraint match on the first letter in the perceptual span. We call this mechanism a *disjunctive retrieval* capability. Except for this last compromise, the disjunctive retrieval capability retrieves the same lexical item as a soft constraint retrieval. Even with this last compromise, computation of activations is slower than real-time in the worst case where only a first letter match is required, since there may be thousands of matching lexical items whose activation must be computed. We are looking for ways to improve processing with minimal compromise compared to the theoretically preferred soft constraint retrieval mechanism.

The verification step is also problematic from an ACT-R modeling perspective. ACT-R does not provide the kind of low level perceptual matching capability that is needed to implement this step. Instead, we have incorporated the Levenshtein Distance algorithm to perform this comparison. We view verification as a key element of the word recognition mechanism in accord with the Activation-Verification model of Paap et al. (1987) and in contrast to the Interactive-Activation model of McClelland & Rumelhart (1985) which has no verification stage. Verification is crucial for identifying novel inputs. A novel input is one that is not a close match to any chunk in memory. Exactly what constitutes a "close match" is an open research question. A side effect of verification is the need to support direct access to two chunks—i.e. the retrieved chunk and the chunk which encodes the perceptual input—in order to perform verification. ACT-R's single chunk buffer constraint is problematic for verification, and for comparison of chunks, more generally.

Multi-Level Activation Spread

In ACT-R, activation spreads from the slots in buffers to chunks in DM with matching slot values. For example, we have a context buffer that encodes information about the context that has a slot named "gram-pos-bias" (i.e. grammatical part of speech bias). Following the processing of a word like "the" (i.e. a determiner), this slot will be set to the symbolic value "noun". When this slot has the symbolic value, it spreads activation to all lexical items in DM with a slot that has the same symbolic value "noun", during a retrieval. If the word "point" follows "the", this bias will spread activation to the noun entry for "point" as opposed to the verb entry (i.e. "to point"). In this way the grammatical context biases the selection of the part of speech of a word during retrieval.

There is no mechanism in ACT-R to spread activation from slots in chunks in DM to other chunks in DM with matching slot values. Once activation spreads from slots in buffers to DM chunks during a retrieval, activation spread stops and the final activation is computed to determine which DM chunk to retrieve. We refer to this as *single level activation spread*.

We are in the process of mapping the linguistic representations that are generated by our language analysis model into a situation model based semantic representation. We are trying to do this in a representationally reasonable way within ACT-R. The problem we face is the many-to-many mapping between words and concepts. Individual words may map to multiple concepts (river "bank" vs. financial "bank"), and individual concepts may map to multiple words ("dog" vs. "canine"). Given this many-to-many mapping, we would like to use mapping chunks to map from words to concepts. The mapping chunks would encode a single mapping relationship (e.g. a separate mapping chunk to map from the word "bank" to the financial institution concept; from the word "bank" to the river bank concept; from the concept dog to the word "dog"; from the concept dog to the word "canine"). When processing a word, a key goal is to retrieve the contextually relevant concept. We would like to accomplish this in a single retrieval, however, we do not know how to do this given the single-level activation spreading mechanism in ACT-R in combination with mapping chunks. Since there is no direct link between a word and a concept if mapping chunks are used (i.e. there is no slot in the concept that contains the word), the word will not spread activation to the concept. Instead, given the use of mapping chunks, two retrievals are needed: 1) given the word, retrieve a mapping chunk, and 2) given a mapping chunk, retrieve a concept. Since our model of language analysis is already slower than adult humans at processing language, any extra retrievals are problematic. In fact, we have already eliminated an extra retrieval in determining the part-of-speech of a word. Previously, two retrievals were needed: 1) retrieve the word form (or multi-word unit) corresponding to the perceptual input, and 2) given the word form (and context) retrieve the part-of-speech of the word. While we were successful in eliminating a retrieval, the resulting word-pos (i.e. word form part of speech) chunks contain a mixture of word form information (e.g. the letters and trigrams in the word) and part of speech (POS) information. Note that this mixture of word form and POS information makes it possible to capture the interaction of word form and POS with single level activation spread. For example, retrieval of the POS

for “speed” (i.e. noun or verb) given the input "spped" depends on the biasing context (e.g. noun bias following “the”) as well as the letters and trigrams. However, the word-pos chunks do not yet contain any representation of phonetic, phonemic, syllabic or morphemic information. With just letter, trigram and POS information, long words contain many filled slots. Adding phonetic, phonemic, syllabic and morphemic information will increase the number of slots substantially. Ideally, we would like to represent letter, trigram, POS, phonetic, syllabic etc. information independently of each other in separate chunks—allowing them to interact in retrieving a word (or letter, or POS, or phoneme), but given the single-level activation spreading mechanism in ACT-R, there is no way to capture the interaction without including all the linguistic knowledge in a single chunk.

The situation is even worse when we consider the mapping from words to concepts. The fall back for mapping words to concepts is to embed all the possible concepts as slot values in complex word-pos-concept chunks (i.e. word form + pos + all matching concepts), or to have multiple word-pos-concept chunks—one for each form, POS, concept combination. While we consider these alternatives to be representationally problematic—we do not know how else to work around the single-level activation spread in ACT-R without introducing extra retrievals. Even with extra retrievals, the interactions between word form, pos and concept will be lost if the word form, part of speech and concept chunks are kept distinct. Further, since each retrieval requires a minimum of 50 msec for the retrieval production to execute, plus the retrieval time, separate retrieval of concepts will add to the mismatch between the model and adult reading rates.

The primary empirical argument against the need for multi-level activation spread in ACT-R is based on studies which show no activation from words like "bull" to words like "milk", even though "bull" activates "cow" and "cow" activates "milk" (ref). Even if it is true that there are no instances of *indirect* activation from "bull" to "milk" (although the empirical evidence is equivocal), this does not rule out the need for multi-level activation spread. There is a hidden assumption that "cow" and "bull" are directly associated, and that "cow" and "milk" are also directly associated. Such direct associations may seem reasonable in small-scale models addressing specific spreading activation phenomena, but they are questionable in a larger-scale model. Do we really want to include all the direct associates of "cow" as slot values in the "cow" chunk (or have multiple “cow” chunks, one for each associate), and do the same for all other chunks?

We understand that the inclusion of a multi-level activation spreading mechanism in ACT-R would be computationally expensive. However, we would like to have the capability to explore use of such a mechanism and to look for ways to keep it computationally tractable. One approach would be to limit the number of chunks from which activation can spread to other chunks within DM. For example, in terms of the mapping from words to concepts, perhaps only the mapping chunks can pass thru activation in DM.

Carry Over Activation and Resonance

Activations are computed in ACT-R as part of a retrieval attempt. The activation computation involves combining the base level activation and activation spread from all buffers which are sources of activation. The logarithmic nature of the base level activation computation means that the base level of overused DM chunks does not vary much from use to use (i.e. the base level activation has reached asymptote). Words constitute very highly used DM chunks. Using estimates of the number of occurrences of a word over a lifetime results in a base level activation that varies little from use to use and decays very slowly. The basic result is that the use of a word does not change the base level activation significantly. Since the spread of activation is computed independently on each retrieval, for a word that has been used recently, there is no contextual indication of this prior use (i.e. the base level hasn’t significantly changed and any prior spread of activation is not retained). Yet there is clear evidence of priming effects from prior uses of words. To overcome this problem we have had to reduce the numbers of uses that are used to compute base level activations so that the use of a word causes a detectable change in base level activation. An alternative is to retain the word in the context buffer so that activation can continue to spread from the word to the corresponding chunk in DM. We have tried this latter approach in the case of idiom processing.

To see the basic challenge, consider the processing of idioms like “kicked the bucket” and verb-particle combinations like “pick…up” as in “pick the ball up”. We assume that idioms and verb-particles correspond to distinct chunks (i.e. multi-word units) in DM. These multi-word expressions exceed the size of the perceptual span and cannot be recognized in a single attention fixation. Instead, the model must somehow recognize the idiom when the word “bucket” is being processed and the verb-particle combination with the word “up” is processed. If there is no evidence that “kicked” has occurred at the processing of “bucket”, then there is no way for the model to retrieve “kicked the bucket” instead of “bucket”. Similarly for “pick” when “up” is processed. Since the DM element “bucket” is an exact match to “bucket” and “bucket” has a higher base frequency than “kicked the bucket” (i.e. single words have a higher base frequency than multi-word units containing them), there must be some mechanism for preferring “kicked the bucket” in this context. “Kicked” and “the” could be retained in the context to spread activation to “kicked the bucket” to handle this example, but, in general, this would mean retaining an arbitrary number of words in the context to spread activation. In the case of “pick…up”, “pick” would need to be retained in a buffer for an indefinite period of time (e.g. “*pick* the big red ball *up*”, “*pick* the ball that is on the table *up*”).

Relying on an increase in base level for "kicked the bucket" will not work in this example. Since the processing of "kicked" is likely to retrieve "kicked" and not "kicked the bucket", the base level activation of "kicked the bucket" will be unaffected at the processing of "kicked" (i.e. the "kicked the bucket" chunk must be retrieved and merged back into DM to constitute a use). Further, any temporary spreading activation from "kicked" to "kicked the bucket" will have been lost at the processing of "bucket".

A possible solution is to introduce a carry-over activation capability. When "kicked" is processed it will spread activation to "kicked the bucket" as well as "kicked". Despite the fact that "kicked the bucket" is not retrieved, some of this activation will carry-over so that when "bucket" is processed, "kicked the bucket" will receive activation from "bucket" as well as carry-over activation from "kicked" and "the". The combination of carry-over activation from "kicked" and "the", plus the activation from "bucket" should allow "kicked the bucket" to be retrieved in this context. In general, this seems like a better solution than trying to retain "kicked" and "the" in the context when "bucket" is processed. In the case of "pick…up", carry over activation should handle cases where the gap between "pick" and "up" is small (e.g. "*pick* the ball *up*"), but cause problems when the gap is large enough that any carry over activation will have decayed. This result might explain the preference for placing the particle before the object when the description of the object is long (e.g. "*pick up* the big red ball on the table" is preferred over "*pick* the big red ball on the table *up*").

There are additional reasons for suggesting the introduction of carry-over activation. Carry-over activation corresponds to a short-term increase in the activation of a DM chunk that extends beyond the execution of a single chunk retrieval. This carry-over activation differs from increases in base level activation which we view as more permanent changes in long-term potentiation. The introduction of carry-over activation combined with multi-level activation spread, could support an ART-like adaptive resonance capability (Grossberg, 1987)—although it is unclear how this could be done in a computationally tractable way. Note that ART uses resonance to distinguish novel from previously experienced inputs—previous inputs lead to resonance with memory whereas novel inputs do not. With carry-over activation, the verification stage of the word recognition subcomponent could be implemented within the architecture via resonance instead of using the Levenshtein Distance metric outside the architecture. The introduction of carry-over activation (and resonance) might also provide an ACT-R equivalent to the long-term working memory (LTWM) of Ericsson & Kintsch (1995) (see discussion of LTWM below).

Inhibition

Inhibition is a winner-take-all mechanism that is commonly used in connectionist architectures to allow a network of nodes to settle into a solution (cf. McClelland & Rumelhart, 1981; Kintsch, 1998). There is no equivalent in ACT-R—although it is possible to get inhibitory effects by explicitly setting the strength of association between two or more chunks to a negative value. But even here, there is no notion of settling into a solution in ACT-R.

The need for inhibition as a mechanism for settling into a solution is obviated in ACT-R by the retrieval mechanism which results in selection of the single most highly activated chunk matching the retrieval template. This is ACT-R's equivalent of a "winner-take-all" network. The retrieval mechanism picks out the most highly activated chunk at a particular level of representation as specified in the retrieval template. For example, the language analysis model has productions that retrieve word-pos (i.e. word form + POS) chunks based on activation from letters, trigrams and POS biases. Of course, a different production could retrieve the most highly activated letter or trigram, if the current task required it (e.g. a word superiority experiment demonstrating that letters are more rapidly recognized in the context of words than in isolation).

ACT-R's spreading activation mechanism doesn't bias a model to any particular task. The same cannot be said of inhibition. For any inhibitory network, it is possible to define conflicting tasks that the inhibitory network cannot perform. For example, if both singular and plural forms of nouns (e.g. "child" and "children") occur in a network, should they inhibit each other? It depends on the task. If the task is a lexical decision task, then we want "child" to inhibit "children" and vice versa, so that they don't interfere (i.e. if "child" is the winner when the input is "children", presumably the lexical decision response would be negative since "child" doesn't match the input). On the other hand, if the task is to generate the singular form of the word in response to the plural word, or the plural in response to the singular, than we need facilitation rather than inhibition. As another example, consider the verbs "go" and "went". For a lexical decision task, these words should inhibit each other. But for a task of generating the past tense of "go", they should facilitate each other. And what about the recognition of multi-word expressions like "get up"? If "get" and "up" inhibit each other, how do we recognize "get up" as a unit?

Not only are inhibitory links task specific, but in a large declarative memory, the number of such links will be explosive. If a word must inhibit all its competitors, with a 61,000+ word lexicon, the number of inhibitory links is computationally explosive. Inhibitory links don't scale. In sum, inhibition is not a viable alternative to ACT-R's task general spreading activation mechanism combined with a task specific retrieval mechanism (i.e. productions effecting retrievals are task specific).

Conclusions

The use of ACT-R for language analysis provides several benefits. ACT-R solves the problems of how to integrate symbolic and probabilistic processing combined with serial

and parallel processing in an effective and elegant manner. For the most part, the capabilities provided by ACT-R have proved useful for the development of our language analysis model, and much of the success of our model is attributable to the capabilities and constraints of ACT-R. I actually believe that adherence to well established cognitive constraints like those provided by ACT-R may facilitate the development of functional language processing systems. Humans are the best and perhaps only example of a system capable of understanding natural language. In an earlier attempt to implement a language analysis system in Prolog (Ball, 1992), it became clear that the lack of any probabilistic mechanisms, and the combination of serial processing and algorithmic backtracking that Prolog provides, were inadequate to deal with the rampant non-determinism of natural language—despite the linguistic roots of Prolog and its widespread use in language processing. Without probabilistic mechanisms to guide language analysis, and with strictly serial processing and algorithmic backtracking, the non-determinism of natural language is overwhelming. A more recent paper by Urszkovitz (2002) reveals that the problems of non-determinism continue to plaque language analysis systems which attempt to do deep (as opposed to superficial statistical) analysis of the linguistic input.

ACT-R is a big improvement over Prolog as a language for natural language analysis. But there is room for improvement of ACT-R as well. The biggest challenge is to figure out how to extend ACT-R's capabilities in the ways suggested in this paper in a computationally tractable manner.

References

Altmann, G. (1998). Ambiguity in sentence processing. *Trends in Cognitive Sciences*, *2*(4), 146-152.

Altmann, G., & Steedman, M. (1988). Interaction with context during human sentence processing. *Cognition*, *30*, 191-238.

Anderson, J. (2007). *How Can the Human Mind Occur in the Physical Universe?* NY: Oxford University Press.

Baddeley, A. (2000). The episodic buffer: a new component of working memory? *Trends in Cognitive Sciences, 4(11),* 417-423.

Ball, J. (1992). PM, Propositional Model, a Computational Psycholinguistic Model of Language Comprehension Based on a Relational Analysis of Written English. Ann Arbor, MI: UMI Dissertation Information Service.

Ball, J. (2004). A Cognitively Plausible Model of Language Comprehension. *Proceedings of the 13th Conference on Behavior Representation in Modeling and Simulation*, pp. 305-316. ISBN: 1-930638-35-3

Ball, J. (2007a). A Bi-Polar Theory of Nominal and Clause Structure and Function. *Annual Review of Cognitive Linguistics*, 27-54. Amsterdam: John Benjamins.

Ball, J. (2007). Construction-Driven Language Processing. *Proceedings of the 2nd European Cognitive Science Conference*, 722-727. Edited by S. Vosniadou, D. Kayser & A. Protopapas. NY: LEA.

Ball, J. (2008). A Naturalistic, Functional Approach to Modeling Language Comprehension. *Papers from the AAAI Fall 2008 Symposium, Naturally Inspired Artificial Intelligence*. Menlo Park, CA: AAAI Press

Ball, J. (2011). A Pseudo-Deterministic Model of Human Language Processing. *Proceedings of the 2011 Cognitive Science Society Conference*.

Ball, J., Freiman, M., Rodgers, S. & Myers, C. (2010). Toward a Functional Model of Human Language Processing. *Proceedings of the 32nd Annual Meeting of the Cognitive Science Society.*

Ball, J., Heiberg, A. & Silber, R. (2007). Toward a Large-Scale Model of Language Comprehension in ACT-R 6. In R. Lewis, T. Polk & J. Laird (Eds.) *Proceedings of the 8th International Conference on Cognitive Modeling*. 173-179. NY: Psychology Press.

Bever, T. (1970). The cognitive basis for linguistic structures. In J. Hayes (Ed.), *Cognition and the development of language* (pp. 279-362). New York: Wiley.

Bothell, D. (2010). The ACT-R 6.0 Reference Manual. Downloadable at http://act-r.psy.cmu.edu/actr6/reference-manual.pdf

Ericsson, K. & Kintsch, W. (1995). Long-term working memory. *Psychological Review, 201,* 211-245.

Freiman, M. & Ball, J. (2010). Improving the Reading Rate of Double-R-Language. In D. D. Salvucci & G. Gunzelmann (Eds.), *Proceedings of the 10th International Conference on Cognitive Modeling* (pp. 1-6). Philadelphia, PA: Drexel University.

Gibson, E., & Pearlmutter, N. (1998). Constraints on sentence comprehension. *Trends in Cognitive Sciences*, *2(*7), 262-268.

Grossberg, S. (1987). Competitive Learning: From Interactive Activation to Adaptive Resonance. *Cognitive Science*, 22, 23-63.

Heiberg, A., Harris, J. & Ball, J. (2007). Dynamic Visualization of ACT-R Declarative Memory Structure. *Proceedings of the 8th International Conference on Cognitive Modeling*, 243-244. Edited by R. Lewis, T. Polk & J. Laird. NY: Psychology Press.

Just, M. & Carpenter, P. (1987). *The Psychology of Reading and Language Comprehension*. Boston: Allyn and Bacon, Inc.

Kintsch, W. (1998). *Comprehension: A paradigm for cognition*. NY: Cambridge University Press.

McClelland, J., & Rumelhart, D. (1981). An interactive activation model of context effects in letter perception: I. An account of basic findings. *Psychological Review*, *88*(5), 375-407.

Paap, K., Newsome, S., McDonald, J., & Schvaneveldt, R. (1982). An Activation-Verification Model of Letter and Word Recognition: The Word-Superiority Effect. *Psychological Review*, *89*, 573-594.

Tanenhaus, M., Spivey-Knowlton, M., Eberhard, K., & Sedivy, J. (1995). Integration of visual and linguistic information in spoken language comprehension. *Science, 268*(5217),1632-1634.

Uszkoreit, H. (2002). New Chances for Deep Linguistic Processing. *Proceedings of COLING 2002*. Taipei.