Unit 5: Activation and Context

The goal of this unit is to introduce the components of the activation equation that reflect the context of a declarative memory retrieval.

5.1 Spreading Activation

The first context issue we will consider is called spreading activation. The chunks in the slots of the chunk in the goal buffer provide a context in which to perform a retrieval. Those chunks spread activation to the other chunks in declarative memory based on their relations to the other chunks, which we call their strengths of association. This essentially results in increasing the activation of those chunks which are related to the current context.

The equation for the activation A_i of a chunk i including spreading activation is defined as:

$$A_i = B_i + \sum_j W_j S_{ji} + \varepsilon$$

Measures of Prior Learning, B_i: The base-level activation reflects the recency and frequency of practice of the chunk as described in the previous unit.

Sources of Activation: The elements j being summed over are the chunks which are in the slots of the goal chunk.

Weighting: W_j is the amount of activation from source j.

Strengths of Association: S_{ji} is the strength of association from source j to chunk i.

ε: The noise value as described in the last unit.

The weights, W_j, of the activation spread defaults to an even distribution. The total amount of source activation is called W and defaults to 1 where it is usually left, though it can be set as the goal activation global parameter (:ga in the sgp command). The W_j values are then set to W/n where n is the number of chunks in goal slots.
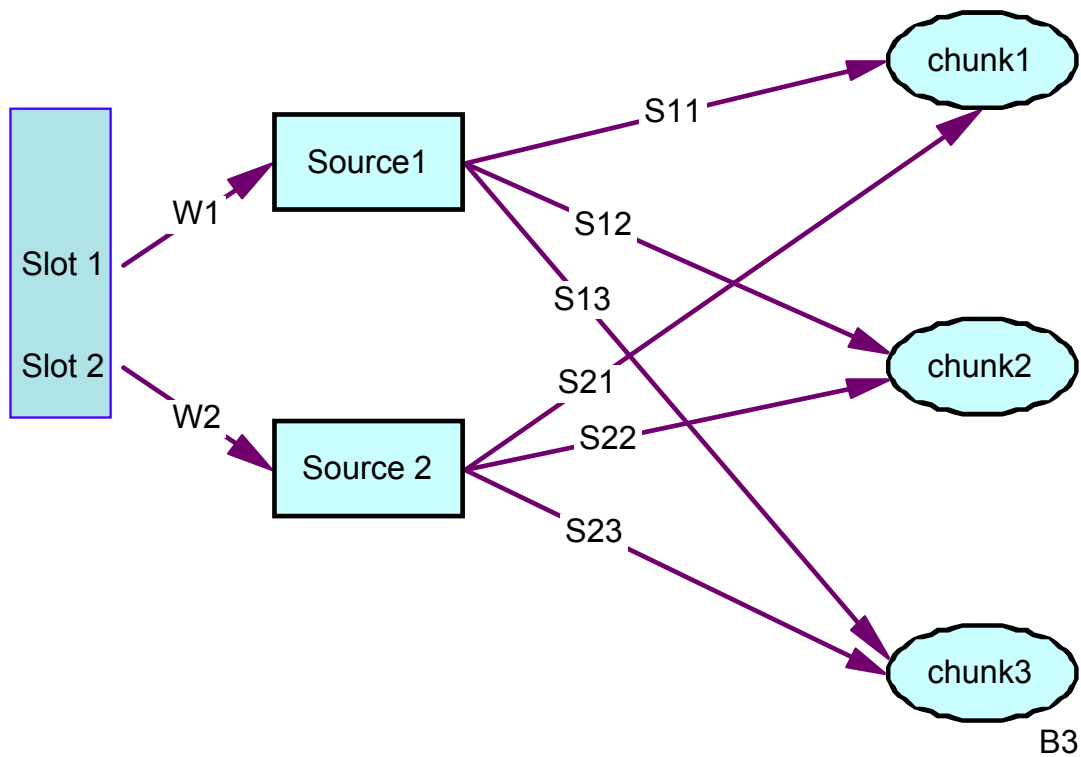
The strength of association, S_{ji}, between two chunks is 0 if chunk j is not in a slot of chunk i or is not itself chunk j and is set using this equation when chunk j is in a slot of chunk i or is itself chunk j:

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

S: A parameter to be estimated (set with the maximum associative strength, :mas, parameter)

fan_j: is the number of chunks in which j is the value of a slot plus one for chunk j being associated with itself.

Here is a diagram to help you visualize how the spreading activation works. Consider a goal chunk that has two chunks in its slots when a retrieval is requested and that there are three chunks which match the retrieval request for which the activation needs to be determined.

The activations of the three chunks are then:

$$A_1 = B_1 + W_1 S_{11} + W_2 S_{21}$$
$$A_2 = B_2 + W_1 S_{12} + W_2 S_{22}$$
$$A_3 = B_3 + W_1 S_{13} + W_2 S_{23}$$

5.2 The Fan Effect

Anderson (1974) performed an experiment in which participants studied 26 facts such as the following sentences:

```
 1. A hippie is in the park.
 2. A hippie is in the church.
 3. A hippie is in the bank.
 4. A captain is in the park.
 5. A captain is in the cave.
 6. A debutante is in the bank.
 7. A fireman is in the park.
 8. A giant is in the beach.
 9. A giant is in the dungeon.
10. A giant is in the castle.
11. A earl is in the castle.
12. A earl is in the forest.
13. A lawyer is in the store.
…
```

After studying these facts, they had to judge whether they saw facts such as the following:

```
A hippie is in the park.
A hippie is in the cave.
A lawyer is in the store.
A lawyer is in the park.
A debutante is in the bank.
A debutante is in the cave.
A captain is in the bank.
```

which contained both studied sentences (targets) and new sentences (foils).

The people and locations for the study sentences could occur in any of one, two, or three of the study sentences. That is called their fan. The following tables show the recognition latencies in seconds for targets and foils as a function of person fan and location fan:

	Targets				Foils			
Location	Person Fan				Person Fan			
Fan	1	2	3	Mean	1	2	3	Mean
1	1.111	1.174	1.222	1.169	1.197	1.221	1.264	1.227
2	1.167	1.198	1.222	1.196	1.250	1.356	1.291	1.299
3	1.153	1.233	1.357	1.248	1.262	1.471	1.465	1.399
Mean	1.144	1.202	1.357	1.20	1.236	1.349	1.340	1.308

The main effects in the data are that as the fan increases the time to respond increases and that foil sentences take longer to respond to than the targets. We will now show how these effects can be modeled using spreading activation.

5.3 Fan Effect Model

The **fan** model in the unit 5 materials contains a model for the testing phase of the experiment. The study portion of the task is not included for simplicity and the model already has chunks defined that encode all of the studied sentences and it can perform one trial of the testing phase. Here is a trace of the model performing one trial for the target sentence "The lawyer is in the store":

```
> (test-sentence-model "lawyer" "store" t 'person)
 Time  0.000: Vision found LOC4349
 Time  0.000: Find-Person Selected
 Time  0.050: Find-Person Fired
 Time  0.050: Module :VISION running command FIND-LOCATION
 Time  0.050: Vision found LOC4350
 Time  0.050: Attend-Person Selected
 Time  0.100: Attend-Person Fired
 Time  0.100: Module :VISION running command MOVE-ATTENTION
 Time  0.185: Module :VISION running command ENCODING-COMPLETE
 Time  0.185: Vision sees TEXT4344
 Time  0.185: Retrieve-Person Selected
 Sources of activation are: (Retrieving-Person)
 CHUNK Lawyer Activation 10.000 Latency  0.000
 Time  0.235: Retrieve-Person Fired
 Time  0.235: Lawyer Retrieved
 Time  0.235: Encode-Person Selected
 Time  0.285: Encode-Person Fired
 Time  0.285: Module :VISION running command FIND-LOCATION
 Time  0.285: Vision found LOC4352
 Time  0.285: Attend-Location Selected
 Time  0.335: Attend-Location Fired
 Time  0.335: Module :VISION running command MOVE-ATTENTION
 Time  0.420: Module :VISION running command ENCODING-COMPLETE
 Time  0.420: Vision sees TEXT4343
 Time  0.420: Retrieve-Location Selected
 Sources of activation are: (Lawyer Retrieving-Location)
 CHUNK Store Activation 10.000 Latency  0.000
 Time  0.470: Retrieve-Location Fired
 Time  0.470: Store Retrieved
 Time  0.470: Encode-Location Selected
 Time  0.520: Encode-Location Fired
 Time  0.520: Retrieve-From-Person Selected
 Sources of activation are: (Lawyer Store)
    Spreading activation  0.453 from source Store level  0.500 times IA  0.907
    Spreading activation  0.453 from source Lawyer level  0.500 times IA  0.907
 CHUNK P13 Activation  0.907 Latency  0.254
 Time  0.570: Retrieve-From-Person Fired
 Time  0.824: P13 Retrieved
 Time  0.824: Yes Selected
 Time  0.874: Yes Fired
 Time  0.874: Module :MOTOR running command PRESS-KEY
 Time  1.024: Module :MOTOR running command PREPARATION-COMPLETE
 Time  1.074: Module :MOTOR running command INITIATION-COMPLETE
 Time  1.084: Device running command OUTPUT-KEY

<< Window "Sentence Experiment" got key #\k at time 1084 >>

 Time  1.174: Module :MOTOR running command FINISH-MOVEMENT
 Time  1.174: Checking for silent events.
 Time  1.174: * Nothing to run:  No productions, no events.
```

To run the model through one trial of the test phase you can call the function **test-sentence-model**. It takes four parameters. The first is a string of the person for the probe sentence. The second is a string of the location for the probe sentence. The third is whether the correct answer is true (t) or false (nil), and the last is either 'person or 'location to choose which of the retrieval productions is used (more on that later). The model can be run over each of the conditions to produce a data fit using the **average-person-location** function:

```
> (average-person-location)
CORRELATION:  0.864
MEAN DEVIATION:  0.049

TARGETS:
                        Person fan
  Location      1             2             3
    fan
    1        1.084 (T  )   1.142 (T  )   1.190 (T  )
    2        1.142 (T  )   1.212 (T  )   1.271 (T  )
    3        1.190 (T  )   1.271 (T  )   1.339 (T  )

FOILS:
    1        1.230 (T  )   1.275 (T  )   1.313 (T  )
    2        1.275 (T  )   1.320 (T  )   1.358 (T  )
    3        1.313 (T  )   1.358 (T  )   1.396 (T  )
```

Two parameters were estimated to produce that fit to the data. They are the latency factor, F, set to .63 and the maximum associative strength, S, set to 1.6. We will now look at how this model performs the task and how spreading activation leads to the effects in the data.

5.4 Model Representations

The study sentences are encoded in comprehend-sentence chunks like this:

```
(p13 isa comprehend-sentence
     state study
     relation in
     arg1 lawyer
     arg2 store)
```

which are propositions encoding the result of past study in the form of an association among the concepts (in this case in, lawyer, and store for "The lawyer is in the store").

There are also meaning chunks which connect the text read from the display to the concepts. For instance, relevant to the case above we have

```
    (lawyer isa meaning
            word "lawyer")

    (store  isa meaning
            word "store")
```

The base-level activations of these meaning chunks have been set to 10 to reflect the fact that they are well practiced and should not fail to be retrieved while the activations of the comprehend-sentence chunks are left at the default of 0.

5.5 Perceptual Encoding

In this section we will briefly describe the productions that perform the perceptual portion of the trial. This is similar to the steps that have been done in previous models and thus it should be fairly familiar.

The entire sentence is presented on the screen, but the model only reads the person and location words from the display to perform the task. If the model were to read all of the words in the sentence it would be difficult to be able to respond fast enough to match the experimental data, and in fact recent studies of the fan effect done using an eye tracker verify that in general participants only fixate those two words from the sentences during the testing trials. To make this easier to handle for the model the sentences are presented with the words in fixed locations on the display. To read and encode the words the model goes through a four step process.

The first production to fire issues a request to the **visual-location** buffer to find the person word:

```
(P find-person
   =goal>
      ISA           comprehend-sentence
      arg1          nil
      state         test
==>
   =goal>
      state         looking-for-person
   +visual-location>
      ISA           visual-location
      screen-x      (within 105 135)
)
```


Although the text for the word always starts at the same location its exact position will vary based on the length of the word, and thus a within test is used to specify where that word should be found.

The next production harvests that location and requests a shift of attention to it:

```
(P attend-person
   =goal>
      ISA           comprehend-sentence
      state         looking-for-person
   =visual-location>
      ISA           visual-location
   =visual-state>
      ISA           module-state
      modality      free
==>
```

```
=goal>
   state       attending-person
+visual>
   ISA         visual-object
   screen-pos  =visual-location
)
```

Then the chunk in the **visual** buffer is harvested and a retrieval request is made to request the chunk that represents the meaning of that word:

```
(P retrieve-person
   =goal>
      ISA         comprehend-sentence
      state       attending-person
   =visual>
      ISA         text
      value       =word
==>
   =goal>
      state       retrieving-person
   +retrieval>
      ISA         meaning
      word        =word
)
```

Finally, that retrieval request is harvested and the meaning chunk is placed into a slot of the goal:

```
(P encode-person
   =goal>
      ISA         comprehend-sentence
      state       retrieving-person
   =retrieval>
      ISA         meaning
==>
   =goal>
      arg1        =retrieval
      state       looking-for-location
   +visual-location>
      ISA         visual-location
      screen-x    (within 400 430)
)
```

This production then issues the **visual-location** request to find the location word and a similar set of productions attend and encode the information for that word.

5.4 Determining the Response

After the encoding has happened the goal chunk will look like this for the sentence "The lawyer is in the store.":

```
Goal
   isa COMPREHEND-SENTENCE
   relation nil
   arg1 Lawyer
   arg2 Store
   state Get-Retrieval
```

and then one of these two productions will be selected and fired to retrieve a study sentence:

```
(P retrieve-from-person
   =goal>
      ISA           comprehend-sentence
      arg1          =person
      state         get-retrieval
==>
   =goal>
      state         nil
   +retrieval>
      ISA           comprehend-sentence
      arg1          =person
      state         study
)

(P retrieve-from-location
   =goal>
      ISA           comprehend-sentence
      arg2          =location
      state         get-retrieval
==>
   =goal>
      state         nil
   +retrieval>
      ISA           comprehend-sentence
      arg2          =location
      state         study
)
```

A thorough model of the task would have those two productions competing and one would randomly be selected. However, to simplify things for demonstration the experiment code forces one or the other to be selected for each trial. The data is averaged over two runs of each trial – one instance of each of those productions being selected.

One important thing to notice is that those productions request the retrieval of a studied chunk based only on one of the items from the probe sentence. By doing so it ensures that one of the study sentences will be retrieved instead of a failure for the event of a foil. If retrieval failure were used by the model to detect the foils then there would be no difference in response times for the foil probes because the time to fail is based solely upon the retrieval threshold, but the data clearly shows that the fan of the items affects the time to respond to foils.

After one of those productions fires a comprehend sentence chunk representing a study trial will be retrieved and one of the following productions will fire to produce a response:

```
(P yes
   =goal>
      ISA           comprehend-sentence
      arg1          =person
      arg2          =location
      state         nil
   =retrieval>
      ISA           comprehend-sentence
      arg1          =person
      arg2          =location
      state         study
==>
   =goal>
      state         done
   +manual>
      ISA           press-key
      key           "k"
)

(P mismatch-person
   =goal>
      ISA           comprehend-sentence
      arg1          =person
      arg2          =location
      state         nil
   =retrieval>
      ISA           comprehend-sentence
   -  arg1          =person
==>
   =goal>
      state         done
   +manual>
      ISA           press-key
      key           "d"
)

(P mismatch-location
   =goal>
      ISA           comprehend-sentence
      arg1          =person
      arg2          =location
      state         nil
   =retrieval>
      ISA           comprehend-sentence
   -  arg2          =location
==>
   =goal>
      state         done
   +manual>
      ISA           press-key
      key           "d"
)
```

If the retrieved sentence matches the probe then the model responds with the true response, "k", and if either one of the components does not match then the model responds with "d".

5.5 Retrieving the Critical Study Chunk

The perceptual encoding productions have a fixed cost of 520 ms and the time to respond after retrieving a comprehend-sentence chunk is 260 ms. Those times are constant across all trials.   The difference in the conditions will result from the times to retrieve the studied sentences. Recall from the last unit that the time to retrieve a chunk i is based on its activation and specified by the equation:

$$Time_i = Fe^{-A_i}$$

Thus, it is differences in activation of the comprehend-sentence chunks which will result in the different times to respond to different trials.

The goal chunk at the time of the retrieval (after either retrieve-from-person or retrieve-from-location fires) will look like this:

```
Goal
    isa COMPREHEND-SENTENCE
    relation nil
    arg1 person
    arg2 location
    state nil
```

where *person* and *location* will be the chunks that represent the meanings for the particular probe being presented. The retrieval request will look like:

```
+retrieval>
    isa  comprehend-sentence
    arg1 person
    state study
```

or this:

```
+retrieval>
    isa  comprehend-sentence
    arg2 location
    state study
```

depending on which of the productions was chosen to perform the retrieval.

The important thing to note is that because the sources of activation in the goal are the same for either retrieval request the spreading activation will not differ between the two cases. You might wonder then why we would need to have both options. That will be described in the detailed examples below.

5.5.1 A simple target trial

The first case we will look at is the target sentence "The lawyer is in the store". Both the person and location in this sentence have a fan of one – they each only occur in that one study sentence.

Thus, the goal chunk will look like this at the time of the critical retrieval:

```
Goal
    isa COMPREHEND-SENTENCE
    relation nil
    arg1 lawyer
    arg2 store
    state nil
```

We will now look at the retrieval that results from the retrieve-from-person production firing. For the following traces we have enabled the activation trace parameter (:act) and the exact match trace (:emt) parameter which show a lot of additional information when a retrieval attempt is made (the :emt parameter is disabled in the model as provided because it generates a lot of additional trace material which is not really important). Here is the trace of the model when that retrieval occurs:

```
 Time  0.520: Retrieve-From-Person Selected
 Sources of activation are: (Lawyer Store)
 Matching CHUNKs of type COMPREHEND-SENTENCE.
 Matching CHUNK P1.
 Value Hippie is different from condition Lawyer in slot arg1: test fails.
 Matching CHUNK P2.
 Value Hippie is different from condition Lawyer in slot arg1: test fails.
 Matching CHUNK P3.
 Value Hippie is different from condition Lawyer in slot arg1: test fails.
 Matching CHUNK P4.
 Value Captain is different from condition Lawyer in slot arg1: test fails.
 Matching CHUNK P5.
 Value Captain is different from condition Lawyer in slot arg1: test fails.
 Matching CHUNK P6.
 Value Debutante is different from condition Lawyer in slot arg1: test fails.
 Matching CHUNK P7.
 Value Fireman is different from condition Lawyer in slot arg1: test fails.
 Matching CHUNK P8.
 Value Giant is different from condition Lawyer in slot arg1: test fails.
 Matching CHUNK P9.
 Value Giant is different from condition Lawyer in slot arg1: test fails.
 Matching CHUNK P10.
 Value Giant is different from condition Lawyer in slot arg1: test fails.
 Matching CHUNK P11.
 Value Earl is different from condition Lawyer in slot arg1: test fails.
 Matching CHUNK P12.
 Value Earl is different from condition Lawyer in slot arg1: test fails.
 Matching CHUNK P13.
    Spreading activation  0.453 from source Store level  0.500 times IA  0.907
    Spreading activation  0.453 from source Lawyer level  0.500 times IA  0.907
 Activation  0.907 is larger than previous best  0.000: selecting P13.
 Matching CHUNK Goal.
```

```
Value nil is different from condition Study in slot state: test fails.
CHUNK P13 Activation  0.907 Latency  0.254
Time  0.570: Retrieve-From-Person Fired
```

With those traces on it shows the chunks that are the sources of activation – lawyer and store. Then it shows all of the chunks that are attempted to be matched to the retrieval request and the activation that is spread from the source to those chunks.

In this case, the only chunk which matches the request is chunk p13 and it is also the only one to receive any spreading activation. Note that this would look exactly the same if the retrieve-from-location production had fired because it would still be the only chunk that matched the request and the sources of activation are the same regardless of which one fires.

This portion of the trace shows the equations we have described above being computed by the model:

```
Matching CHUNK P13.
    Spreading activation  0.453 from source Store level  0.500 times IA  0.907
    Spreading activation  0.453 from source Lawyer level  0.500 times IA  0.907
 Activation  0.907 is larger than previous best  0.000: selecting P13.
…
 CHUNK P13 Activation  0.907 Latency  0.254
```

Remember that we have set the parameter F to .63, the parameter S to 1.6, and the base-level activation for the chunks is 0 in this model.

Looking at this trace, we see the S_{ji} values from store to P13 and lawyer to P13 (called the IA in the trace) are both .907. That comes from the equation:

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

The value of S was estimated to fit the data as 1.6 and the fan of both the store and lawyer chunks is 2. They each occur as a slot value in only the P13 chunk plus one for their reference to themselves. Then substituting into the equation we get:

$$S_{(store)(p13)} = S_{(lawyer)(p13)} = 1.6 - \ln(2) = 0.907$$

The W_j values are .5 because the source activation is the default value of 1 and there are two source chunks. Thus the activation of chunk P13 is:

$$A_i = B_i + \sum_j W_j S_{ji}$$

12

$$A_{p13} = 0 + (.5*.907) + (.5*.907) = .907$$

Finally, we see the time to complete the retrieval (the latency listing in the trace) computed as:

$$Time_i = Fe^{-A_i}$$

$$Time_{p13} = .63e^{-.907} = 0.254$$

Adding that retrieval time to the fixed costs of .52 seconds to do the perception, .05 seconds to fire the production that requests the retrieval, and .26 seconds to perform the response gives us a total of 1.084 seconds, which the value in the fan 1-1 cell of the model data presented above.

Now that we have looked at the intricate details of how the retrieval times are determined for the simple case we will look at a few other cases.

5.5.2 A different target trial

The target sentence "The hippie is in the bank" is a more interesting case to look at. Hippie is the person in three of the study sentences and bank is the location in two of them. Now we will see why it takes the model longer to respond to such a probe. Here are the critical components from the trace when retrieve-from-person is chosen:

```
Time  0.520: Retrieve-From-Person Selected
Sources of activation are: (Hippie Bank)
Matching CHUNKs of type COMPREHEND-SENTENCE.
Matching CHUNK P1.
   Spreading activation  0.107 from source Hippie level  0.500 times IA  0.214
Activation  0.107 is larger than previous best  0.000: selecting P1.
Matching CHUNK P2.
   Spreading activation  0.107 from source Hippie level  0.500 times IA  0.214
Activation  0.107 is larger than previous best  0.107: selecting P2.
Matching CHUNK P3.
   Spreading activation  0.251 from source Bank level  0.500 times IA  0.501
   Spreading activation  0.107 from source Hippie level  0.500 times IA  0.214
Activation  0.358 is larger than previous best  0.107: selecting P3.
…
CHUNK P3 Activation  0.358 Latency  0.441
```

There are three chunks that match the request for a comprehend-sentence chunk with an arg1 value of hippie. Each receives the same amount of activation being spread from hippie. Because hippie is a member of three chunks it has a fan of 4 and thus the $S_{(hippie)i}$ value is:

$$S_{(hippie)i} = 1.6 - \ln(4) = 0.214$$

Chunk P3 also contains the chunk bank in its arg2 slot and thus receives the source spreading from it as well.

Now we will look at the case when retrieve-from-location fires for this probe sentence:

```
Time  0.520: Retrieve-From-Location Selected
Sources of activation are: (Hippie Bank)
Matching CHUNKs of type COMPREHEND-SENTENCE.
…
Matching CHUNK P3.
   Spreading activation  0.251 from source Bank level  0.500 times IA  0.501
   Spreading activation  0.107 from source Hippie level  0.500 times IA  0.214
Activation  0.358 is larger than previous best  0.000: selecting P3.
Matching CHUNK P6.
   Spreading activation  0.251 from source Bank level  0.500 times IA  0.501
…
CHUNK P3 Activation  0.358 Latency  0.441
```

In this case there are only two chunks which match the request for a comprehend-sentence chunk with an arg2 value of bank.

Regardless of which production fired to request the retrieval, chunk P3 had the highest activation because it received spreading activation from both sources. Thus, even if there is more than one chunk which matches the retrieval request issued by retrieve-from-person or retrieve-from-location the correct study sentence will always be retrieved because its activation will be the highest, and that activation value will be the same in both cases.

Notice that the activation of P3 is less than the activation that P13 had in the previous example because the source activation being spread is less. That is because the sources here have a higher fan, and thus a lesser S_{ji}. Because the activation is smaller, it takes longer to retrieve such a fact and that gives us the difference in response time effect of fan in the data.

5.5.3 A foil trial

Now we will look at a foil trial. The foil probe "The giant is in the bank" is similar to the target that we looked at in the last section. The person has a fan of three and the location has a fan of two. This time however there is no matching study sentence. Here are the critical components from the trace when retrieve-from-person is chosen:

```
Time  0.520: Retrieve-From-Person Selected
Sources of activation are: (Giant Bank)
Matching CHUNKs of type COMPREHEND-SENTENCE.
…
Matching CHUNK P8.
   Spreading activation  0.107 from source Giant level  0.500 times IA  0.214
Activation  0.107 is larger than previous best  0.000: selecting P8.
Matching CHUNK P9.
   Spreading activation  0.107 from source Giant level  0.500 times IA  0.214
Activation  0.107 is larger than previous best  0.107: selecting P9.
```

```
Matching CHUNK P10.
    Spreading activation  0.107 from source Giant level  0.500 times IA  0.214
Activation  0.107 is larger than previous best  0.107: selecting P10.
…
CHUNK P10 Activation  0.107 Latency  0.566
```

There are three chunks that match the request for a comprehend-sentence chunk with an arg1 value of giant and each receives the same amount of activation being spread from giant. However, none contain an arg2 value of bank. Thus they only get activation spread from one source and have a lesser activation value than the corresponding target sentence probe has. Because the activation is smaller, the retrieval time is greater. This results in the effect of foil trials taking longer than target trials.

Before concluding this section however let us look at the trace if retrieve-from-location were to fire for this foil:

```
Time  0.520: Retrieve-From-Location Selected
Sources of activation are: (Giant Bank)
Matching CHUNKs of type COMPREHEND-SENTENCE.
…
Matching CHUNK P3.
    Spreading activation  0.251 from source Bank level  0.500 times IA  0.501
Activation  0.251 is larger than previous best  0.000: selecting P3.
…
Matching CHUNK P6.
    Spreading activation  0.251 from source Bank level  0.500 times IA  0.501
Activation  0.251 is larger than previous best  0.251: selecting P6.
…
CHUNK P6 Activation  0.251 Latency  0.490
```

In this case there are only two chunks which match the request for a comprehend-sentence chunk with an arg2 value of bank.. Again, the activation of the chunk retrieved is less than the corresponding target trial, but it is not the same as when retrieve-from-person fired. That is why the model is run with each of those productions fired once for each probe and the results averaged together. Otherwise the foil data would only show the effect of fan for the item that was used to retrieve the study chunk.

5.6 Partial Matching

Up to now models have either always retrieved a chunk which matched the retrieval request or resulted in a failure to retrieve anything. Now we will look at modeling errors in recall in more detail. There are two kinds of errors that can occur. One is an error of commission when the wrong thing is recalled. This will occur when the activation of the wrong chunk is greater than the activation of the correct chunk. The second is an error of omission when nothing is recalled. This will occur when no chunk has activation above the retrieval threshold.

We will continue to look at productions from the fan model for now. In particular, this production requests the retrieval of a chunk:

```
(P retrieve-from-person
    =goal>
       ISA          comprehend-sentence
       arg1         =person
       state        get-retrieval
==>
    =goal>
       state        nil
    +retrieval>
       ISA          comprehend-sentence
       arg1         =person
       state        study
)
```

In this case an attempt is being made to retrieve a comprehend-sentence chunk with a particular person (bound to =**person**) that had been studied. If =**person** were the chunk giant, this retrieval request would be looking for a chunk of the form:

```
+retrieval>
      isa comprehend-sentence
      arg1 giant
      state study
```

As was shown above, there were three chunks which matched that request in the study set and one of those will be retrieved.

However, let us consider the case where there had been no study sentences with the person giant but there had been a sentence with the person titan in the location being probed with giant i.e. there was a study sentence "The titan is in the bank" and the test sentence is now "The giant is in the bank". In this situation one might expect that real participants might incorrectly classify the probe sentence as one that was studied because of the similarity between the words giant and titan. The model however could not make such an error.

This is what the partial matching mechanism is designed to address.   When partial matching is enabled (by setting the :pm parameter to t) the similarity between the chunks in the retrieval specification and the chunks in the slots of the chunks in declarative memory are taken into consideration. The chunk with the highest activation will still be the one retrieved, but with partial matching enabled that chunk may not have the exact slot values as specified in the retrieval request.

The activation A_i of a chunk i is defined fully as:

$$A_i = B_i + \sum_j W_j S_{ji} + \sum_k PM_{ki} + \varepsilon$$

B_i, W_j, S_{ji}, and ε have been discussed previously. The new term is the partial matching component.

Specification elements k: The matching summation is computed over the slot values of the retrieval specification.

Match Scale, *P*: This reflects the amount of weighting given to the similarity in slot k. By default this is a constant across all slots with the value of 1 and is set with the :mp parameter.

Match Similarities, M_{ki}: The similarity between the value k in the retrieval specification and the value in the corresponding slot of chunk i.

Similarity values, the M_{ki}'s, can be set by the modeler along with the scale on which they are defined. The scale range is set with a maximum similarity (set using the :ms parameter) and a maximum difference (set using the :md parameter). By default :ms is 0 and :md is -1. The similarity between anything and itself is automatically set to the maximum similarity and by default the similarity between any other pair of values is the maximum difference. However, it is possible to choose some value between maximum similarity and maximum difference for items that are deemed to be somewhat similar. Thus, unless one sets specific similarities or changes the range of similarities, the match similarity will be 0 when the element in the chunk's slot matches the retrieval specification and −1 when it does not.

5.7 Grouped Recall

To illustrate these ideas we will use a model called **grouped**. This is a demonstration model of a grouped recall task which is based on a larger model of a complex recall experiment. As with the **fan** model, the studied items are already specified in the model, so it does not model the encoding and study of the items. In addition, the response times and error profiles of this model are not fit to any data. This demonstration model is designed to show the mechanism of partial matching and how it can lead to errors of commission and errors of omission. Because the model is not fit to any data, and the mechanism being studied does not rely on any of the perceptual or motor modules of ACT-R, they are not being used, and instead the chunks are manipulated "directly" by the Lisp code that provides the experiment. This technique of using only the cognitive system in ACT-R can be useful when modeling a task where the timing is not important or other situations where accounting for a "real world" interaction is not necessary to accomplish the objectives of the model. The experiment description text for this unit gives the details of how that is accomplished in this model.

If you check the global parameters you will see that it has a retrieval threshold of -0.5 and a value of .15 for the transient noise s parameter. Also, to simplify the demonstration the spreading activation described above is disabled by setting the goal activation to 0. This model is set up to recall a list of nine items which are encoded in groups of three elements. The list that should be recalled is (123) (456) (789). To run the model, call the **run-grouped-recall** function. You will get a trace like the one below and the list of responses will be returned. Because of the activation noise it will likely vary from run to run. In this run, notice that it mis-ordered the recall of the 5 and 6 and failed to recall the last item, producing (123) (465) (78).

```
Time  0.000: Recall-First-Group Selected
Time  0.050: Recall-First-Group Fired
Time  0.851: Group1 Retrieved
Time  0.851: Start-Recall-Of-Group Selected
Time  0.901: Start-Recall-Of-Group Fired
Time  1.868: Item1 Retrieved
Time  1.868: Harvest-First-Item Selected
Time  1.918: Harvest-First-Item Fired
Time  2.676: Item2 Retrieved
Time  2.676: Harvest-Second-Item Selected
Time  2.726: Harvest-Second-Item Fired
Time  3.648: Item3 Retrieved
Time  3.648: Harvest-Third-Item Selected
Time  3.698: Harvest-Third-Item Fired
Time  5.347: Failure Retrieved
Time  5.347: Second-Group Selected
Time  5.397: Second-Group Fired
Time  6.471: Group2 Retrieved
Time  6.471: Start-Recall-Of-Group Selected
Time  6.521: Start-Recall-Of-Group Fired
Time  7.443: Item4 Retrieved
Time  7.443: Harvest-First-Item Selected
Time  7.493: Harvest-First-Item Fired
Time  8.528: Item6 Retrieved
Time  8.528: Harvest-Second-Item Selected
Time  8.578: Harvest-Second-Item Fired
Time  9.724: Item5 Retrieved
Time  9.724: Harvest-Third-Item Selected
Time  9.774: Harvest-Third-Item Fired
Time 11.423: Failure Retrieved
Time 11.423: Third-Group Selected
Time 11.473: Third-Group Fired
Time 12.433: Group3 Retrieved
Time 12.433: Start-Recall-Of-Group Selected
Time 12.483: Start-Recall-Of-Group Fired
Time 13.338: Item7 Retrieved
Time 13.338: Harvest-First-Item Selected
Time 13.388: Harvest-First-Item Fired
Time 13.969: Item8 Retrieved
Time 13.969: Harvest-Second-Item Selected
Time 14.019: Harvest-Second-Item Fired
Time 15.668: Failure Retrieved
Time 15.668: Checking for silent events.
Time 15.668: * Nothing to run:  No productions, no events.
("1" "2" "3" "4" "6" "5" "7" "8")
```

5.8 Error of Commission

If one turns on the activation trace and the partial matching trace (set the :act and :pmt parameters to t) one will get a much more exhaustive trace explaining the computations taking place. The following is from the detailed trace of the above run relevant to an error of commission when ACT-R recalls 6 in the second position where 5 is the correct item. The critical comparison is between **item5**, which should be retrieved and **item6**, which is retrieved:

```
 Sources of activation are: (List Group2 Recalling-Item Second)
 Matching CHUNKs of type ITEM.
    Adding noise  0.360
...
    Adding noise -0.241
 Partial matching chunk Item5 with activation -0.241
 Similarity between chunks Group2 and Group2 is  0.000
 Adjusting activation by  0.000 to -0.241
 Similarity between chunks Second and Second is  0.000
 Adjusting activation by  0.000 to -0.241
 Similarity between chunks Unused and Retrieved is -1.000
 Matching score of chunk Item5 is -0.241.
 Activation -0.241 is larger than previous best -0.500: selecting Item5.
    Adding noise  0.465
 Partial matching chunk Item6 with activation  0.465
 Similarity between chunks Group2 and Group2 is  0.000
 Adjusting activation by  0.000 to  0.465
 Similarity between chunks Third and Second is -0.500
 Adjusting activation by -0.500 to -0.035
 Similarity between chunks Unused and Retrieved is -1.000
 Matching score of chunk Item6 is -0.035.
 Activation -0.035 is larger than previous best -0.241: selecting Item6.
```

These report the calculations of the activation equation (ignoring associative strength because **W** is set to 0).

$$A_i = B_i + \sum_k P M_{ki} + \varepsilon$$

In these examples the base-level activations, **B$_i$**, have their default value of 0, the match scale, **P**, has the default value of 1, and the only noise value is the transient component. So the calculations are really just a matter of adding up the match similarities, **M$_{ki}$** and adding the transient noise. The matching of **item5** is repeated below. While the base level is 0 it starts out at -0.241 because of transient noise. The retrieval pattern is

```
+retrieval>
   isa      item
   parent   Group2
   position second
   - status retrieved
```

and item5 is:

```
Item5
    isa ITEM
    name "5"
    parent Group2
    position Second
    status unused
```

which matches both on the parent and position slots resulting in the addition 0 to the base-level activation. It also satisfies the negative test on the status slot and thus does not have to pay the penalty of a mismatch. Thus its final activation is -0.241.

```
    Adding noise -0.241
 Partial matching chunk Item5 with activation -0.241
 Similarity between chunks Group2 and Group2 is  0.000
 Adjusting activation by  0.000 to -0.241
 Similarity between chunks Second and Second is  0.000
 Adjusting activation by  0.000 to -0.241
 Similarity between chunks Unused and Retrieved is -1.000
 Matching score of chunk Item5 is -0.241.
 Activation -0.241 is larger than previous best -0.500: selecting Item5.
```

This activation value is higher than the threshold of -.5, so **item5** is a temporary choice but then we come to the matching of **item6**:

```
Item6
    isa ITEM
    name "6"
    parent Group2
    position Third
    status unused


    Adding noise  0.465
 Partial matching chunk Item6 with activation  0.465
 Similarity between chunks Group2 and Group2 is  0.000
 Adjusting activation by  0.000 to  0.465
 Similarity between chunks Third and Second is -0.500
 Adjusting activation by -0.500 to -0.035
 Similarity between chunks Unused and Retrieved is -1.000
 Matching score of chunk Item6 is -0.035.
 Activation -0.035 is larger than previous best -0.241: selecting Item6.
```

The similarity between second and third is -0.5. However, the initial activation starts out at 0.465 because of transient noise and so the final activation is 0.465 + -0.5 = -0.035 which is greater than the activation for **item5**. Thus, this wrong item is selected because of the fluctuation in activations.

The similarities between the different positions are defined in the model using the **setsimilarities** command:

```
(setsimilarities
 (first second -0.5)
 (second third -0.5)
 (first third -1))
```

Similarity values are symmetric, thus it is not necessary to also specify (second first -0.5). The similarity between a chunk and itself has the value of maximum similarity by default, and thus also does not need to be specified.

5.9 Error of Omission

Here is the portion of the detailed trace relevant to the failure to recall the ninth item:

```
Sources of activation are: (List Group3 Recalling-Item Third)
 Matching CHUNKs of type ITEM.
...
    Adding noise  0.301
 Partial matching chunk Item8 with activation  0.301
 Similarity between chunks Group3 and Group3 is  0.000
 Adjusting activation by  0.000 to  0.301
 Similarity between chunks Second and Third is -0.500
 Adjusting activation by -0.500 to -0.199
 Similarity between chunks Retrieved and Retrieved is  0.000
 Adjusting activation by -1.000 to -1.199
 Matching score of chunk Item8 is -1.199.
    Adding noise -0.594
 Partial matching chunk Item9 with activation -0.594
 Similarity between chunks Group3 and Group3 is  0.000
 Adjusting activation by  0.000 to -0.594
 Similarity between chunks Third and Third is  0.000
 Adjusting activation by  0.000 to -0.594
 Similarity between chunks Unused and Retrieved is -1.000
 Matching score of chunk Item9 is -0.594.
 CHUNK failure Activation -0.500 Latency  1.649
 No chunk reached activation threshold -0.500: matching fails.
```

Item9 starts out with an activation of –0.594 and because it does not get any penalties added that is its activation, but it is below the threshold of -0.5, so it is not retrieved.

One other thing to note about that trace is the activation of **item8**. It starts out with an activation of 0.301 and gets a penalty of -0.500 based on the similarity between second and third. So, at that point it has an activation of -0.199 which is above the threshold. However, it matches perfectly on the status slot, but the test is one of negation so it also gets the maximum difference penalty added to it. If the retrieval specification did not request an item that had not been previously retrieved **item8** would have been incorrectly retrieved for the ninth position instead of a complete retrieval failure. The important thing about a negative test is that it will not change the activation if the similarity between the chunks in the test is less than the maximum similarity, but if the similarity between the chunks is the maximum similarity it will receive an activation penalty of the maximum difference, which, with the default values is -1.0.

5.10 Unit Exercise: Simple Addition

The following are data obtained by Siegler and Shrager on the relative frequencies of different responses by 4 year olds to addition problems. It seems likely that many of the kids did not know the answers to the larger problems so we will only focus on the addition table from 1+1 to 3+3.

```
        0    1    2    3    4    5    6    7    8   Other (includes no response)
1+1    -   .05  .86   -   .02   -   .02   -    -    .06
1+2    -   .04  .07  .75  .04   -   .02   -    -    .09
1+3    -   .02   -   .10  .75  .05  .01  .03   -    .06
2+2  .02    -   .04  .05  .80  .04   -   .05   -     -
2+3    -     -   .07  .09  .25  .45  .08  .01  .01  .06
3+3  .04    -    -   .05  .21  .09  .48   -   .02  .11
```

The **siegler** model contains the functions to perform a version of this task with a model. As with the **grouped** model, there is no interface generated for the task so it is not possible to run yourself through the experiment, but one would guess that you would probably not make the same mistakes as 4 year olds anyway. For the model, the goal chunk is directly modified to provide the stimuli. This simplification is done because we are not concerned with matching latencies so the time of the encoding is not a factor and also because it is assumed that the errors are not a result of improper encoding of the items.

The function called **test-fact** will present 1 trial with the specified numbers and return the contents of the goal chunk's answer slot, which should be a string from the name slot of a number chunk if the model has an answer. The numbers that are passed to test-fact need to be the names of the numbers as strings. Thus, to run a trial of 1+2 you would call **(test-fact "one" "two")**. When the model is doing the task it is reset before each trial, then the arg1 and arg2 slots of the goal chunk (named goal) are set to the names of the numbers being presented. The model then needs to encode those names and determine a response. The name of that number should be placed in the answer slot of the goal. This is similar to how one would map a visual representation to a corresponding number chunk before attempting a retrieval and then retrieve a chunk to extract it's response value from one of its slots.

The function called **do-one-set** takes no parameters and runs one trial of each problem returning the list of responses.   Since the model is reset each time the order of presentation does not matter and it is not randomized.

The function **run-subjects** takes one parameter which is the number of times to run **do-one-set**. It will then tally all of the responses, report the fit to the data, and display the results in a table.

The following is the output from a run of my model of the task:

```
CORRELATION:  0.976
MEAN DEVIATION:  0.058
        0    1    2    3    4    5    6    7    8   Other
```

```
1+1  0.00  0.07  0.85  0.07  0.00  0.00  0.00  0.00  0.00  0.00
1+2  0.00  0.00  0.06  0.86  0.07  0.00  0.00  0.00  0.00  0.00
1+3  0.00  0.00  0.00  0.09  0.86  0.04  0.00  0.00  0.00  0.01
2+2  0.00  0.00  0.00  0.09  0.90  0.00  0.00  0.00  0.00  0.01
2+3  0.00  0.00  0.00  0.02  0.34  0.55  0.03  0.00  0.00  0.06
3+3  0.00  0.00  0.00  0.00  0.01  0.05  0.70  0.06  0.00  0.17
```

Your task is to write a model that encodes the question presented and then responds with an answer (by placing it in the answer slot of the goal) and produces a distribution of errors that is comparable to the data. This should be achievable by manipulating the similarities, the activation noise, the retrieval threshold, and the base-level activations. The values that these parameters have in the given model are most likely not at the optimal values for fitting the data. The effects that these parameters will have on the data will be described below. Source activation, W, is set to zero and can be left there to do the assignment, but if you would like to explore the effect it has feel free to set W to a non-zero value and experiment.

You are given a set of chunks that encode the numbers from 0-9, the addition table from 0+0 to 5+5, and a goal of type problem. The chunk-type of the goal looks like this:

```
(chunk-type problem arg1 arg2 answer state)
```

The arg1 and arg2 slots are set to contain the string names of the numbers at the beginning of each trial automatically, so your model then needs to encode those so that a plus-fact can (possibly) be retrieved.

The similarities between the number chunks will affect the distribution of incorrect retrievals. While this looks like 45 free parameters to be fit, in practice that is just not reasonable. For a situation like this, where the chunks represent numbers, it is better to set the similarity between two numbers based on the numerical difference between them using a single formula to specify all of them.

To set the similarities you need to use the **setsimilarities** command. Here is an example of its use from the **grouped** model:

```
(setsimilarities
 (first second -0.5)
 (second third -0.5)
 (first third -1))
```

The similarities are symmetric, so the order of the chunks in the lists does not matter, and you only need to set the similarity for the pair once.

The activation noise is going to impact the frequency of incorrect retrievals. The more noise there is the less likely it is that the correct chunk will have the highest activation. The noise is set with the :ans parameter.

The retrieval threshold is going to establish the chance for a retrieval failure. Its setting is going to depend on the average activation levels which will depend on how the other

parameters are set. It may be best to set it to a very low value at the start (like -10) so that there is always a plus-fact retrieved. Then, once the distribution looks good you can increase it until it starts to introduce some failures to improve the fit. It is set using the :rt parameter.

You may also want to set the base-level activation of some chunks directly. Because base-level learning is not enabled, those base-level values will not change as the model runs. This will make particular chunks more likely to be retrieved than others. In the model code provided the base-levels of the number chunks are set high so that there should never be a failure to retrieve a number:

```
(set-general-base-levels
  (zero 10) (one 10) (two 10) (three 10) (four 10) (five 10)
  (six 10) (seven 10)(eight 10) (nine 10))
```

Depending on how you set your other parameters, you may want to adjust the base-level for the numbers even higher than that. This is another instance where it looks like there are a lot of free parameters that could be used to fit the data, but again a principled approach is advised. If you need to adjust the base-levels do it in a manner that seems justified. For example, if +0 facts are more often retrieved as errors than the rest you could set all of those to the same larger base-level. A model that sets a different base-level for each plus-fact does not really demonstrate anything.

There is one other thing that you will need to do to produce the response that has not yet been demonstrated in the tutorial. That is to retrieve a chunk directly. If you have a chunk bound to a variable in a production you can request that that chunk be retrieved like this:

```
(P demo
   =goal>
      ISA          test-goal
      value        =val
==>
   +retrieval>     =val)
```

That will attempt to retrieve that chunk specifically. It does not attempt to retrieve any other chunk, so it will not retrieve a chunk that is similar even if it has a higher activation.

In addition to the parameters suggested, you may also want to adjust the match scale, *P* (set with :mp). Because the default range of similarities is -1.0 to 0.0 there is not going to be much difference in the activations due to similarities between the numbers, and as a result the predictions can be very sensitive to small changes in the parameters. Thus, if you adjust *P* you may find it easier to arrive at a good fit to the data. While this is not essential we suggest changing it to a value like 10 which we used in the reference solution.

Siegler, R. S., & Shrager, J. (1984). Strategy choices in addition and subtraction: How do children know what to do? In C. Sophian (Ed.), Origins of cognitive skills (pp. 229-293). Hillsdale, NJ: Erlbaum.