Performing Computational Contextual Vocabulary Acquisition to Define 'Hysteria'

By Peter Hoellig

CSE663: Advanced Knowledge Representation and Reasoning

December 8th, 2008

Abstract

What does a reader do when they come across a word they do not know and there is no reliable external source of information from which to determine the meaning? They make use of the materials at hand; that is, the textual context which surrounds the word and any appropriate background knowledge that the person possess. The Contextual Vocabulary Acquisition project is an attempt to computationally model this process. Our reader is a SNePS based cognitive agent named CASSIE. We shall make use of the SNePS semantic network based knowledge representation system to represent the passage of text as well as CASSIE's prior background knowledge. Once the background knowledge and the text containing the unknown word are represented, CASSIE can make use of CVA definition algorithms to compute a working definition of the unknown word. This computed definition will then be compared with the results of human experiments with using CVA on the passage as well as to a dictionary definition. What remains to be done with this particular instance of the project will also be discussed.

1. Introduction to SNePS and Contextual Vocabulary Acquisition

Given the dynamic nature of natural language, it is immediately clear that people do not know every single word that they read. Languages evolve over time, new words are invented, old words are discarded, and new meanings for existing words can be created. Therefore, it is unreasonable to believe that there is a complete collection of information about words in a

language. Dictionaries may contain a large amount of information about words, but it is impossible to contain all of the information because new words and meanings can be created at any time. This necessitates the need for a way of determining the meaning of a word without an information source other than the text in which the word appears and the background knowledge of the person reading it.

This is where Contextual Vocabulary Acquisition (CVA) comes in to play. CVA is a method of determining a meaning of a word from the textual context in which it appears combined with the background knowledge of the reader regarding the rest of the text. Since ours is a computational project, our methods of CVA are algorithmic in nature. However, these translate into procedures that can be put into practice by human readers to improve reading comprehension (see Rapaport & Kibby 2008 for more information). Our method of CVA on the computational end is a four step process:

1. Represent enough background knowledge for comprehension of the words and concepts in the passage except for the unknown word.
2. Represent the passage itself so that it may be read by our reader.
3. Have CASSIE (more on her later) read the passage and create a representation of all the information.
4. Execute any applicable word definition algorithms that may apply in order to generate a definition of the word. If there is no applicable algorithm, then execute a

statement that will return all information that has been inferred about the unknown word.

Without CVA or a similar method, a person (or perhaps a computer) is not capable of expanding its knowledge of words in a language without being explicitly informed about the word from an external source of some kind. Without such methods, people would need to constantly consult external sources of information in order to keep up with the evolution of natural language. Clearly, CVA is an important skill for all cognitive agents.

In order for our method of CVA to work, it requires a hefty amount of 'representation'. For this project we make use of the SNePS semantic network knowledge representation and reasoning system. The SNePS user language, SNePSUL, is a language that is syntactically similar to the LISP programming language and is what we use to create all of the representations needed for CVA. The network that results from representing the appropriate background knowledge required for using CVA on the text is an analog for the mind of the reader prior to reading the passage. In our case, that network actually is the mind of our reader prior to reading the passage. The SNePS cognitive agent, CASSIE, makes use of only those facts represented in the network and her built-in definition algorithms for reasoning about the unknown word in the passage.

After reading the passage, CASSIE's mind (the network) looks quite different than it did before she read the passage. It will now contain a representation of the passage itself as well as any inferences that were made from the passage combined with the prior knowledge that

she had before reading the sentence. Applicable definition algorithms are now run which collect information contained in CASSIE's mind and organize it in a concise, dictionary-like fashion. If there are no applicable definitions then it is necessary to query the SNePS system so that CASSIE can tell you what she knows about the previously unknown word.

2. The Passage and the Dictionary

The passage for this project was found in a paper on using context as an aid in reading comprehension:

hysteria:

> She was close to **hysteria**; that is, her voice was slurred, her eyes darted frantically from left to right as if to avoid confronting the scene, and her arms and hands trembled uncontrollably. (Dulin 1970)

The unknown word we have chosen is 'hysteria' and is shown in boldface text.

Because of the difficulties of representing such a long sentence, for the purposes of this project it was broken down into several smaller sentences.

1) She was close to hysteria.

2) She was close to hysteria; that is, her voice was slurred.

3) She was close to hysteria; that is, her eyes darted frantically from left to right as if to avoid confronting the scene.

4) She was close to hysteria; that is, her arms and hands trembled uncontrollable.

Great care must be taken when revising a passage so that nothing is added or subtracted from its meaning. Since our modifications simply change the passage into the several sentences that it would be if there was no use of commas in it, the meaning should certainly be preserved. We also dropped the 'and' in the last section of the sentence since it is a purely grammatical construct and provides no additional information to the reader.

Since our goal is to provide dictionary-like definitions for the reader, we shall provide one here, according to Merriam-Webster's online dictionary:

> 1 **:** a psychoneurosis marked by emotional excitability and disturbances of the psychic, sensory, vasomotor, and visceral functions
> 2 **:** behavior exhibiting overwhelming or unmanageable fear or emotional excess <political *hysteria*> (http://www.merriam-webster.com/dictionary/hysteria)

Both definitions could be the intended meaning, but the first is more likely to be the case for our passage.

3. Verbal Protocols

In this project, we make use of experiments with humans to determine the meaning of the unknown word from context and then use the information from the experiments as the background knowledge for CASSIE. However, we are presented with the problem that 'hysteria' is a fairly common word that the majority of subjects will know the meaning of. In order to prevent prior knowledge of the word itself interfering with the CVA process, in our experiments we substituted a nonexistent word 'jastoie' in its place.

Half a dozen of these experiments were performed and the results centered on a certain type of shock or fear that effects physical ability, there was also one subject who determined that the word was a synonym for intoxication. Subjects generally determined their answers by making use of prior information of what causes the symptoms that are listed. All subjects made use of the idea that 'that is' signals that whatever follows the phrase is meant to clarify what exactly is meant by the words prior to the phrase.

As the results of the verbal protocols dictate, we must focus on the idea that 'that is' signals clarification as well as the potential causes of the symptoms that she is suffering from. Possible causes of a slurred voice include being intoxicated and being afraid of something, trembling can be caused by fear and instability, and desire to avoid something is also a result of fear. These results will make up the focus of our representation of CASSIE's knowledge prior to reading the passage.

4. Background Knowledge

a. Syntax and Semantics

In this project, we will make use of several predefined SNePS case frames to represent our background knowledge. We shall make use of the following:

- Object – Property

- Object1 – Rel – Object2
- Lex
- Agent – Act – Action

The syntax and semantics of these are located at:

http://www.cse.buffalo.edu/~rapaport/CVA/CaseFrames/case-frames/

We will also make use of the forall – ant – cq case frame from the SNePS Case Frame Dictionary at: http://www.cse.buffalo.edu/sneps/Manuals/dictionary.pdf

b. Background Knowledge Representation

Now that we have defined what we need to represent and how we are going to represent it, the time has come to actually do it. The following representations are in SNePSUL and use the syntax and semantics as previously defined. The following is the SNePS representation of CASSIE's knowledge prior to reading the passage.

```
; If someone's voice is slurred then they are scared.
; Unrepresented: Could also mean that they are drunk or have some kind of speech disability.

(describe (assert forall ($person)
    ant(build object *person property (build lex "Voice\ is\ slurred") )
    cq(build object *person property (build lex "Scared") )))
```

In the verbal protocols it was determined that one of the causes of a slurred voice was being scared or fearful. We use the forall – ant – cq and object – property case frames here. The semantics of this statement is “For all people, if a person’s voice is slurred then that person is scared”. This rule like many others is defeasible, there are other reasons that a person’s voice may be slurred, but for our purposes it will do.

```
; If someone is trembling then they are unstable.
(describe (assert forall ($person)
 ant(build object *person property (build lex "Trembling") )
 cq(build object *person property (build lex "Unstable") )
 cq(build object *person property (build lex "Fearful") )))
```

The semantics of this is, “For all people, if a person is trembling then, that person is unstable and fearful”. This is also a rule derived from the verbal protocols and is defeasible as well. It makes use of the forall – ant – cq and object – property case frames.

```
; Thatis(P(x), Q(x)) => P(x) ^ (P(x) => Q(x)) ^ ((P(x) => Q(x)) => P(Q))

(describe (assert forall ($person $prop $prop2)
      ant(build object1 (build object *person property *prop)
            rel (build lex "that\ is")
            object2 (build object *person property *prop2))
      cq(build object *person property *prop)
      cq(build forall () ant(build object *person property *prop)
```

```
cq(build object *person property *prop2)
cq(build object *prop property *prop2))))
```

This rule represents the phrase ‘that is’ in our representation. Its semantics is “If the statement ‘person a has property p’ is followed by ‘that is’ which is followed by ‘person a has property q’ then it is the case that ‘person a has property p’ and it is also the case that ‘person a has property p’ logically implies that ‘person a has property q’ and ‘property p has property q’ “. In more familiar English, if “Sally is dreaming that is, Sally is sleeping” then “Sally is dreaming”, “Sally is dreaming” implies “Sally is sleeping”, and “Sally is dreaming” implies “Sally is sleeping” implies that “dreaming” implies “sleeping”. This is one of several rules that were considered for representing ‘that is’. It was chosen because it appeared to be closer to what people actually did when reasoning than its functionally equivalent counterparts. Among those considered was a rule of the form “Thatis(P(x),Q(x)) => P(x) ^ P(Q) ^ (P(x) => Q(x))”. This was discarded because it requires the inference that, in the previous example “Sally is dreaming that is, Sally is sleeping” implies “dreaming” implies “sleeping”. In the verbal protocols however, it seemed to be the case that subjects inferred P(Q) using a rule of transitivity from P(x) and Q(x) rather than jumping directly to P(Q).

5. CASSIE Reads the Passage

;She was close to hysteria.

(describe (add object #her property (build lex "hysteria")))

Curiously, this metaphorical spatial relation was largely unmentioned in the verbal protocols. Most subjects treated the sentence as if it said “She was hysterical” rather than “She was close to hysteria”. During the work on the project, attempts at representing the metaphorical ‘close to’ always resulted in the need for creating new rules with clear syntax and very awkward (or just plain nonsensical) semantics in order to preserve the definition. It is represented here as “She was hysterical” as in the results of the verbal protocols.

;She was close to hysteria that is; her voice was slurred:

(describe (add object1 (build object *her property (build lex "hysteria"))

rel (build lex "that\ is")

object2 (build object *her property (build lex "Voice\ is\ slurred"))))

We make use of the object1 – rel – object2 case frame for all cases where ‘that is’ appears in the sentences. The first object is the statement of “She was hysterical”, the relation is “that is”, and the second object is “Her voice was slurred” represented as an object – property arc from her to the lexicon entry for “Voice is slurred”.

```
;She was close to hysteria that is;
;Her eyes darted frantically from left to right as if to
;avoid confronting the scene

(describe (add object1 (build object *her property (build lex "hysteria"))
            rel (build lex "that\ is")
            object2 (add agent *her act
                  (build action (build lex "Eyes\ darting\ frantically")))))

(describe (add object1 (build object *her property (build lex "hysteria"))
            rel (build lex "that\ is")
            object2 (add agent *her act
                    (build action (build lex "Avoiding\ the \scene")))))


(describe (add object1 (build object *her property (build lex "hysteria")
                )
            rel (build lex "that\ is")
            object2 (build object *her property (build lex "Frantic") )))
```

This sentence is split up into three statements for ease of understanding on the part of the developer. Splitting “eyes darted frantically as if to avoid confronting the scene” at the “as if” and removing it. These are represented with the standard object1 – rel – object2 arc for use with that is and the statements themselves are represented using the agent – act – action case

frame. Also included is that “she is frantic” which is included because she is described with the adjective frantic.

```
;She was close to hysteria that is; Her arms and hands trembled uncontrollably.
        (describe (add object1 (build object *her property (build lex "hysteria")
            )
          rel (build lex "that\ is")
          object2 (add agent *her act (build action
                (build lex "Arms\ tremble\ uncontrollably")))))

(describe (add object1 (build object *her property (build lex "hysteria")
            )
          rel (build lex "that\ is")
          object2 (add agent *her act (build action
            (build lex "Hands\ tremble\ uncontrollably")))))

(describe (add object1 (build object *her property (build lex "hysteria")
            )
          rel (build lex "that\ is")
          object2 (build object *her property (build lex "Trembling") )))
```

The representation of this sentence is syntactically identical to the last sentence. The object1 – rel – object2 case frame is used for the ‘that is’ relation as is standard. The action of her arms and hands trembling is represented with agent – act – action arcs as before. Also

added is the statement that “She is trembling” because she is described as trembling in the text.

(describe (find object (build lex "hysteria") property ?x))

Because we are ignoring the ‘close to’ relation and operating on ‘She was hysterical’ as per the verbal protocols, we are actually defining ‘hysterical’ rather than ‘hysteria’. This means that we are working with an adjective and thus there is no algorithm for coming up with a definition. Because of this we must formulate a query to the SNePS system so that CASSIE will tell us what she has determined.

(m14! (object (m10 (lex hysteria)))
(property (m1 (lex Voice is slurred))))
(m16! (object (m10)) (property (m2 (lex Scared))))
(m29! (object (m10)) (property (m26 (lex Frantic))))
(m41! (object (m10)) (property (m4 (lex Trembling))))
(m43! (object (m10)) (property (m6 (lex Fearful))))
(m44! (object (m10)) (property (m5 (lex Unstable))))

The above is the response that CASSIE gives to us. It tells us that ‘hysteria’ has the properties of a slurred voice, being scared, being frantic, trembling, and being fearful. Looking

back at the definition, “a psychoneurosis marked by emotional excitability and disturbances of the psychic, sensory, vasomotor, and visceral functions” we have a much less academic definition than the Merriam-Webster. However, what we do have are instances of most of the symptoms of hysteria which is a close approximation of the Merriam-Webster. Also, our definition models the responses to the verbal protocols well and one could hardly expect someone to come up with such a medicinal definition from the passage we were analyzing.

6. Future Work

There remains some work to be done with this particular instance of the project before it could be called complete, as there is one part of the dictionary definition still missing that could be at least in part derived from the sentence. The Merriam-Webster refers to hysteria as a ‘psychoneuroses’ which is something we are lacking. This is contained in the sentence by the metaphorical spatial relation ‘close to’ which we ignored in our representation. Attempts to include this in our representation typically met with failure because when the spatial relation is forced into the ‘that is’ rules the semantics always resulted in incoherency.

Representing the appropriate background knowledge for CASSIE to infer that hysteria is a state of mind is not incredibly difficult. The passage begins, “She was close to hysteria” the standard inference is that hysteria would then be a person or place. However, this ‘close to’ relation is metaphorical and means directly that her state of mind was close to hysteria. It can be logically determined that hysteria is not a person or location because it is not written in the

sentence with a capital letter at its beginning. This could be used to make a non defeasible rule about determining whether or not the spatial 'close to' is being metaphorically used in any given sentence. Also included in the appendix (after the demo run of the background knowledge) of this paper is another run that deduces all of the information presented here as well as that hysteria is a state of mind. However, it does this by using different representations for the statement "She was close to hysteria" at different times and was not included in the paper because of this.

Appendix A – Demo Run

```
nickelback {~/CSE663} > mlisp
;;; Installing locale patch, version 1.
International Allegro CL Enterprise Edition
8.1 [Linux (x86)] (Oct 27, 2008 11:52)
Copyright (C) 1985-2007, Franz Inc., Oakland, CA, USA.  All Rights Reserved.

This development copy of Allegro CL is licensed to:
  [4549] University at Buffalo

;; Optimization settings: safety 1, space 1, speed 1, debug 2.
;; For a complete description of all compiler switches given the
;; current optimization settings evaluate (explain-compiler-settings).
;;---
;; Current reader case mode: :case-sensitive-lower
cl-user(1): :ld /projects/snwiz/bin/sneps
; Loading /projects/snwiz/bin/sneps.lisp
;;; Installing jlinker patch, version 1.
;;; Installing regexp2-s patch, version 1.
Loading system SNePS...10% 20% 30% 40% 50% 60% 70% 80% 90% 100%
SNePS-2.7 [PL:1 2008/02/12 17:19:45] loaded.
Type `(sneps)' or `(snepslog)' to get started.
cl-user(2): (sneps)


  Welcome to SNePS-2.7 [PL:1 2008/02/12 17:19:45]

Copyright (C) 1984--2007 by Research Foundation of
State University of New York. SNePS comes with ABSOLUTELY NO WARRANTY!
Type `(copyright)' for detailed copyright information.
Type `(demo)' for a list of example applications.

  12/1/2008 17:51:45

* (demo "Hysteriathatis.demo")

File /home/unmdue/phoellig/CSE663/Hysteriathatis.demo is now the source of input.


 CPU time : 0.00
```

```
*
;Clears the network.
(resetnet t)

Net reset


 CPU time : 0.00

*
; load the noun definition algorithm
^(
--> load "/projects/rapaport/CVA/STN2/defun_noun.cl")
; Loading /projects/rapaport/CVA/STN2/defun_noun.cl
t



 CPU time : 0.03

*
; load all pre-defined relations
^(
--> load "/projects/rapaport/CVA/STN2/demos/rels")
; Loading /projects/rapaport/CVA/STN2/demos/rels
t



 CPU time : 0.00

*
; load all pre-defined path definitions
^(
--> load "/projects/rapaport/CVA/mkb3.CVA/paths/paths")
; Loading /projects/rapaport/CVA/mkb3.CVA/paths/paths
before implied by the path (compose before
                    (kstar (compose after- ! before)))
before- implied by the path (compose (kstar (compose before- ! after))
                     before-)
after implied by the path (compose after
                   (kstar (compose before- ! after)))
after- implied by the path (compose (kstar (compose after- ! before))
                     after-)
```

```
sub1 implied by the path (compose object1- superclass- ! subclass
                    superclass- ! subclass)
sub1- implied by the path (compose subclass- ! superclass subclass- !
                    superclass object1)
super1 implied by the path (compose superclass subclass- ! superclass
                    object1- ! object2)
super1- implied by the path (compose object2- ! object1 superclass- !
                    subclass superclass-)
superclass implied by the path (or superclass super1)
superclass- implied by the path (or superclass- super1-)
t


 CPU time : 0.00

*



;She was close to hysteria; that is, her voice was slurred, her eyes darted
;frantically from left to right as if to avoid confronting the scene,
;and her arms and hands trembled uncontrollably.




; If someone's voice is slurred then they are scared.
; Unrepresented: Could also mean that they are drunk
; or have some kind of speech disability.

(describe (assert forall ($person)
      ant(build object *person property (build lex "Voice\ is\ slurred") )
      cq(build object *person property (build lex "Scared") )
      )
)

(m3! (forall v1)
 (ant (p1 (object v1) (property (m1 (lex Voice is slurred)))))
 (cq (p2 (object v1) (property (m2 (lex Scared))))))

(m3!)
```

```
 CPU time : 0.00

*
; If someone is trembling then they are unstable.
(describe (assert forall ($person)
       ant(build object *person property (build lex "Trembling") )
       cq(build object *person property (build lex "Unstable") )
       cq(build object *person property (build lex "Fearful") )
       )
)

(m7! (forall v2) (ant (p3 (object v2) (property (m4 (lex Trembling)))))
 (cq (p5 (object v2) (property (m6 (lex Fearful))))
  (p4 (object v2) (property (m5 (lex Unstable))))))

(m7!)

 CPU time : 0.00

*



;that is

; Thatis(P(x), Q(x)) => P(x) ^ (P(x) => Q(x)) ^ ((P(x) => Q(x)) => P(Q))

(describe (assert forall ($person $prop $prop2)
       ant(build object1 (build object *person property *prop)
             rel (build lex "that\ is")
             object2 (build object *person property *prop2)
         )
       cq(build object *person property *prop)
       cq(build forall () ant(build object *person property *prop)
                   cq(build object *person property *prop2)
                   cq(build object *prop property *prop2)

         )
        )
)
```

```
(m9! (forall v5 v4 v3)
 (ant
  (p8 (object1 (p6 (object v3) (property v4)))
   (object2 (p7 (object v3) (property v5))) (rel (m8 (lex that is)))))
 (cq (p10 (ant (p6)) (cq (p9 (object v4) (property v5)) (p7))) (p6)))

(m9!)

 CPU time : 0.00

*
```

```
;She was close to hysteria.
(describe (add object #her property (build lex "hysteria") ) )

(m11! (object b1) (property (m10 (lex hysteria))))

(m11!)

 CPU time : 0.00

*
```

```
;She was close to hysteria that is; her voice was slurred:
(describe (add object1 (build object *her property (build lex "hysteria")
                    )
              rel (build lex "that\ is")
              object2 (build object *her property (build lex "Voice\ is\ slurred"))
      )
)

(m17! (object b1) (property (m2 (lex Scared))))
(m16! (object (m10 (lex hysteria))) (property (m2)))
(m15! (ant (m11! (object b1) (property (m10))))
 (cq (m14! (object (m10)) (property (m1 (lex Voice is slurred))))
  (m12! (object b1) (property (m1)))))
(m13! (object1 (m11!)) (object2 (m12!)) (rel (m8 (lex that is))))

(m17! m16! m15! m14! m13! m12!)
```

```
 CPU time : 0.01

*


;She was close to hysteria that is;
;Her eyes darted frantically from left to right as if to
;avoid confronting the scene

(describe (add object1 (build object *her property (build lex "hysteria")
                  )
            rel (build lex "that\ is")
            object2 (add agent *her act
                     (build action (build lex "Eyes\ darting\ frantically")))
      )
)

(m21! (object1 (m11! (object b1) (property (m10 (lex hysteria)))))
 (object2
  (m20! (act (m19 (action (m18 (lex Eyes darting frantically)))))
   (agent b1)))
 (rel (m8 (lex that is))))

(m21!)

 CPU time : 0.00

*

(describe (add object1 (build object *her property (build lex "hysteria")
                  )
            rel (build lex "that\ is")
            object2 (add agent *her act
                       (build action (build lex "Avoiding\ the \scene")))
      )
)

(m25! (object1 (m11! (object b1) (property (m10 (lex hysteria)))))
 (object2
  (m24! (act (m23 (action (m22 (lex Avoiding the scene)))))
   (agent b1)))
 (rel (m8 (lex that is))))

(m25!)
```

```
 CPU time : 0.01

*




(describe (add object1 (build object *her property (build lex "hysteria")
                    )
               rel (build lex "that\ is")
               object2 (build object *her property (build lex "Frantic") )
       )
)

(m30! (ant (m11! (object b1) (property (m10 (lex hysteria)))))
 (cq (m29! (object (m10)) (property (m26 (lex Frantic))))
  (m27! (object b1) (property (m26)))))
(m28! (object1 (m11!)) (object2 (m27!)) (rel (m8 (lex that is))))

(m30! m29! m28! m27!)

 CPU time : 0.00

*




;She was close to hysteria that is; Her arms and hands trembled uncontrollably.
(describe (add object1 (build object *her property (build lex "hysteria")
                    )
               rel (build lex "that\ is")
               object2 (add agent *her act (build action
                         (build lex "Arms\ tremble\ uncontrollably")))))

(m34! (object1 (m11! (object b1) (property (m10 (lex hysteria)))))
 (object2
  (m33! (act (m32 (action (m31 (lex Arms tremble uncontrollably)))))
   (agent b1)))
 (rel (m8 (lex that is))))

(m34!)

 CPU time : 0.00
```

```
*(describe (add object1 (build object *her property (build lex "hysteria")
                    )
               rel (build lex "that\ is")
               object2 (add agent *her act (build action
                    (build lex "Hands\ tremble\ uncontrollably")))))

(m38! (object1 (m11! (object b1) (property (m10 (lex hysteria)))))
 (object2
  (m37! (act (m36 (action (m35 (lex Hands tremble uncontrollably)))))
   (agent b1)))
 (rel (m8 (lex that is))))

(m38!)

 CPU time : 0.00

*



(describe (add object1 (build object *her property (build lex "hysteria")
                    )
               rel (build lex "that\ is")
               object2 (build object *her property (build lex "Trembling") )
       )
)

(m46! (object b1) (property (m5 (lex Unstable))))
(m45! (object b1) (property (m6 (lex Fearful))))
(m44! (object (m10 (lex hysteria))) (property (m5)))
(m43! (object (m10)) (property (m6)))
(m42! (ant (m11! (object b1) (property (m10))))
 (cq (m41! (object (m10)) (property (m4 (lex Trembling))))
  (m39! (object b1) (property (m4)))))
(m40! (object1 (m11!)) (object2 (m39!)) (rel (m8 (lex that is))))

(m46! m45! m44! m43! m42! m41! m40! m39!)

 CPU time : 0.01

*
```

(describe (find object (build lex "hysteria") property ?x))

(m14! (object (m10 (lex hysteria)))
 (property (m1 (lex Voice is slurred))))
(m16! (object (m10)) (property (m2 (lex Scared))))
(m29! (object (m10)) (property (m26 (lex Frantic))))
(m41! (object (m10)) (property (m4 (lex Trembling))))
(m43! (object (m10)) (property (m6 (lex Fearful))))
(m44! (object (m10)) (property (m5 (lex Unstable))))

(m14! m16! m29! m41! m43! m44!)

 CPU time : 0.00

*

End of /home/unmdue/phoellig/CSE663/Hysteriathatis.demo demonstration.

 CPU time : 0.07

*

Appendix B – Alternate Demo Run

```
nickelback {~/CSE663} > mlisp
;;; Installing locale patch, version 1.
International Allegro CL Enterprise Edition
8.1 [Linux (x86)] (Oct 27, 2008 11:52)
Copyright (C) 1985-2007, Franz Inc., Oakland, CA, USA.  All Rights Reserved.

This development copy of Allegro CL is licensed to:
  [4549] University at Buffalo

;; Optimization settings: safety 1, space 1, speed 1, debug 2.
;; For a complete description of all compiler switches given the
;; current optimization settings evaluate (explain-compiler-settings).
;;---
;; Current reader case mode: :case-sensitive-lower
cl-user(1): :ld /projects/snwiz/bin/sneps
; Loading /projects/snwiz/bin/sneps.lisp
;;; Installing jlinker patch, version 1.
;;; Installing regexp2-s patch, version 1.
Loading system SNePS...10% 20% 30% 40% 50% 60% 70% 80% 90% 100%
SNePS-2.7 [PL:1 2008/02/12 17:19:45] loaded.
Type `(sneps)' or `(snepslog)' to get started.
cl-user(2): (sneps)


  Welcome to SNePS-2.7 [PL:1 2008/02/12 17:19:45]

Copyright (C) 1984--2007 by Research Foundation of
State University of New York. SNePS comes with ABSOLUTELY NO WARRANTY!
Type `(copyright)' for detailed copyright information.
Type `(demo)' for a list of example applications.

  12/1/2008 17:53:16

* (demo "Hysteriastate.demo")

File /home/unmdue/phoellig/CSE663/Hysteriastate.demo is now the source of input.

 CPU time : 0.00

*
```

```
;Clears the network.
(resetnet t)

Net reset


 CPU time : 0.00

*
; load the noun definition algorithm
^(
--> load "/projects/rapaport/CVA/STN2/defun_noun.cl")
; Loading /projects/rapaport/CVA/STN2/defun_noun.cl
t



 CPU time : 0.03

*
; load all pre-defined relations
^(
--> load "/projects/rapaport/CVA/STN2/demos/rels")
; Loading /projects/rapaport/CVA/STN2/demos/rels
t



 CPU time : 0.00

*
; load all pre-defined path definitions
^(
--> load "/projects/rapaport/CVA/mkb3.CVA/paths/paths")
; Loading /projects/rapaport/CVA/mkb3.CVA/paths/paths
before implied by the path (compose before
                     (kstar (compose after- ! before)))
before- implied by the path (compose (kstar (compose before- ! after))
                      before-)
after implied by the path (compose after
                    (kstar (compose before- ! after)))
after- implied by the path (compose (kstar (compose after- ! before))
                     after-)
sub1 implied by the path (compose object1- superclass- ! subclass
```

```
                    superclass- ! subclass)
sub1- implied by the path (compose subclass- ! superclass subclass- !
                    superclass object1)
super1 implied by the path (compose superclass subclass- ! superclass
                    object1- ! object2)
super1- implied by the path (compose object2- ! object1 superclass- !
                    subclass superclass-)
superclass implied by the path (or superclass super1)
superclass- implied by the path (or superclass- super1-)
t



 CPU time : 0.00

*
;     She was close to hysteria; that is, her voice was slurred, her eyes darted
;frantically from left to right as if to avoid confronting the scene, and her arms and
;hands trembled uncontrollably.


;that is
(describe (assert forall ($thing $prop $type)
      &ant((build object *thing property *prop)
            (build member *thing class *type))
      cq(build object *type property *prop)
      )
)

(m1! (forall v3 v2 v1)
 (&ant (p2 (class v3) (member v1)) (p1 (object v1) (property v2)))
 (cq (p3 (object v3) (property v2))))

(m1!)

 CPU time : 0.00

*
(assert member #her class (add lex "close\ to\ hysteria") )

(m3!)

 CPU time : 0.00
```

```
* (add part (add lex "close\ to\ hysteria") whole (add lex "hysteria") )

(m5!)

 CPU time : 0.00

*

; If x is close to y then y is a state of mind.
(describe (assert forall ($thing1 $state)
      &ant((build object1 *thing1 rel (build lex "close\ to")
            object2 *state)
            (build member *state class))
      cq(build member *state class (build lex "state\ of\ mind"))
      )
)

(m8! (forall v5 v4)
 (&ant (p5 (member v5))
  (p4 (object1 v4) (object2 v5) (rel (m6 (lex close to)))))
 (cq (p6 (class (m7 (lex state of mind))) (member v5))))

(m8!)

 CPU time : 0.00

*
; She was close to hysteria.
;(describe (assert object1 #her rel (build lex "close\ to")
;             object2 (add lex "hysteria") ) )


;Her voice was slurred.
(describe(assert object *her property (add lex "Voice\ is\ slurred")
      )
)

(m10! (object b1) (property (m9! (lex Voice is slurred))))

(m10!)

 CPU time : 0.00

*
```

```
;Her eyes darted frantically from left to right as if to
;avoid confronting the scene

(describe (assert agent *her act (build action (add lex "Eyes\ dart\ frantically\ to\ avoid\
confronting\ the\ scene"))))

(m13!
 (act (m12
       (action
        (m11!
         (lex Eyes dart frantically to avoid confronting the scene)))))
 (agent b1))

(m13!)

 CPU time : 0.00

*
;Her arms and hands trembled uncontrollably.

(describe (assert agent *her act (build action (add lex "Arms\ and\ hands\ tremble\
uncontrollably"))))

(m16!
 (act (m15
       (action (m14! (lex Arms and hands tremble uncontrollably)))))
 (agent b1))

(m16!)

 CPU time : 0.00

*

^(
--> defineNoun "hysteria")
 Definition of hysteria:
nil



 CPU time : 0.01
```

*

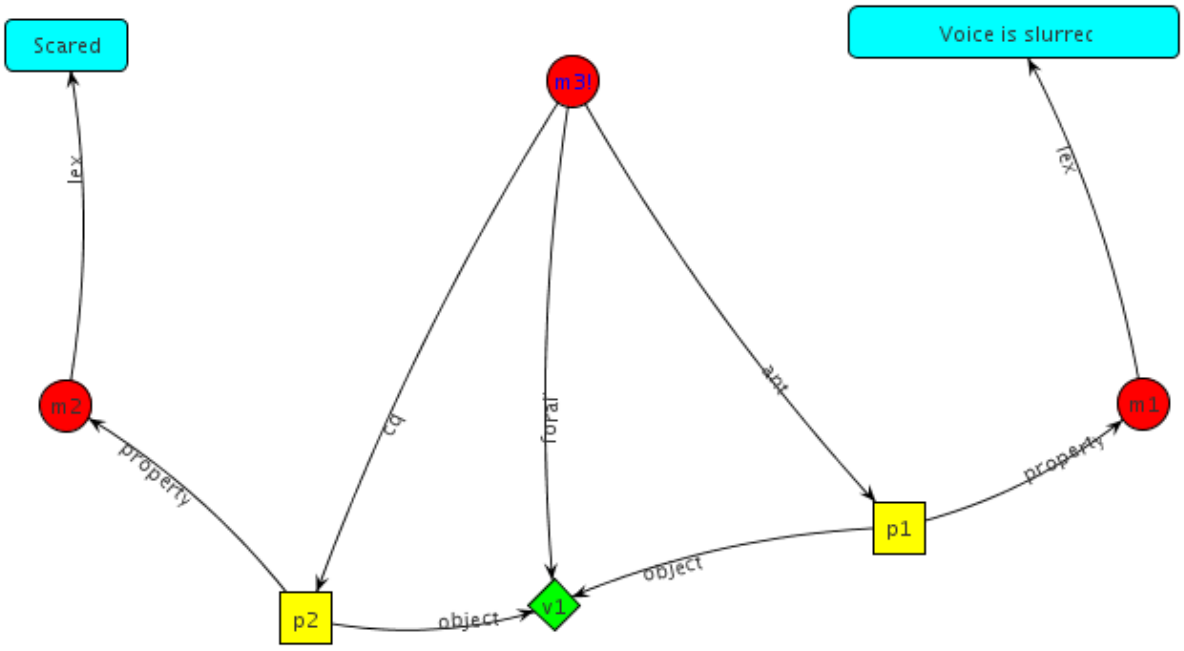
End of /home/unmdue/phoellig/CSE663/Hysteriastate.demo demonstration.

CPU time : 0.05

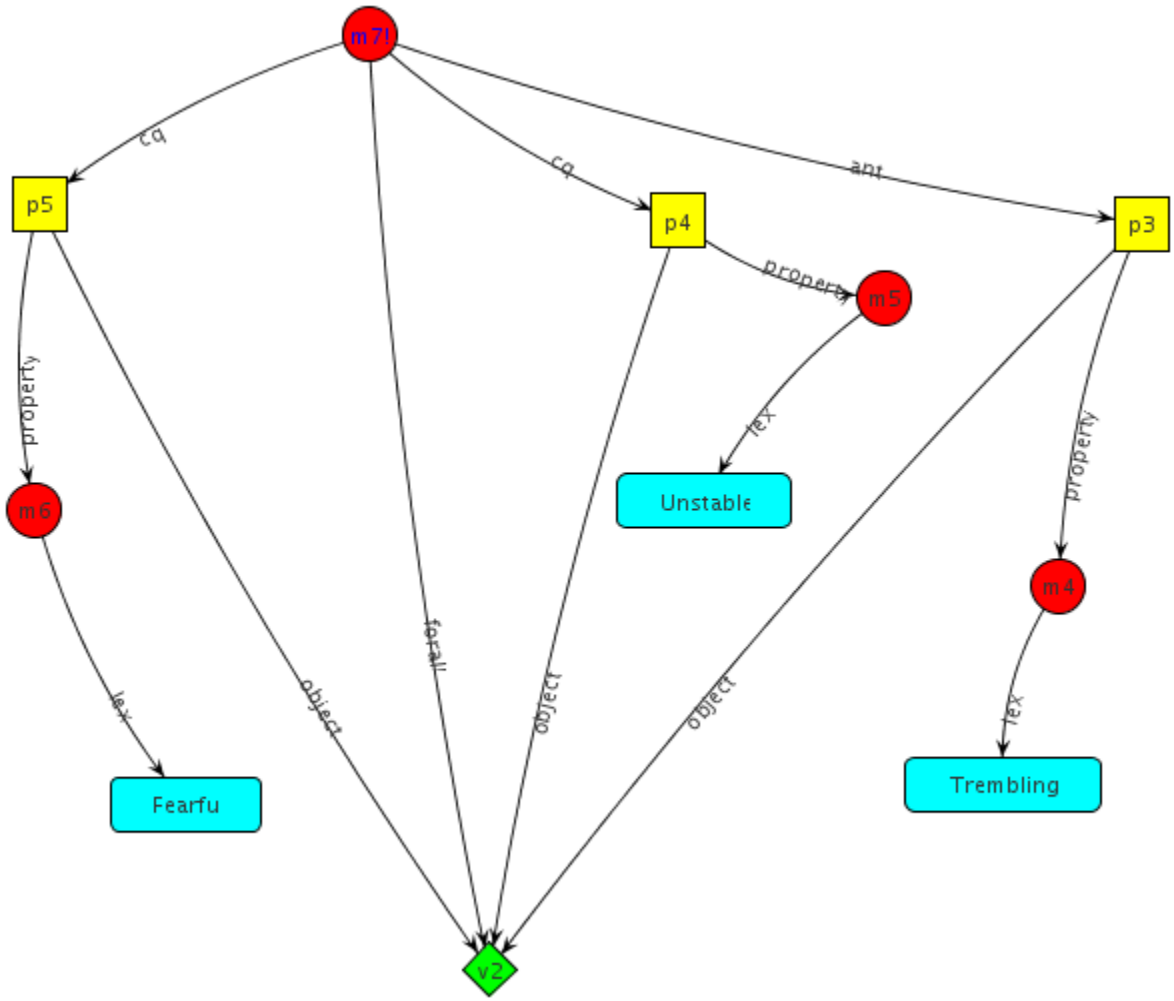
*

Appendix C – Network Descriptions

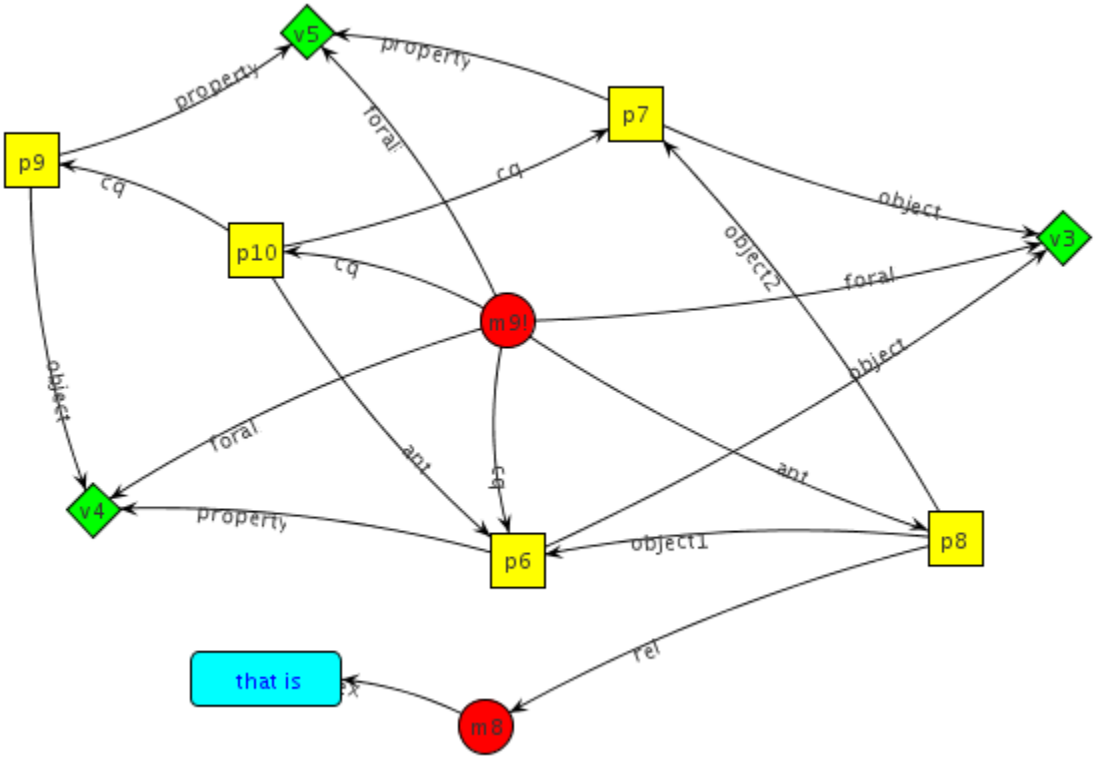
For all x, if x's voice is slurred then x is scared.



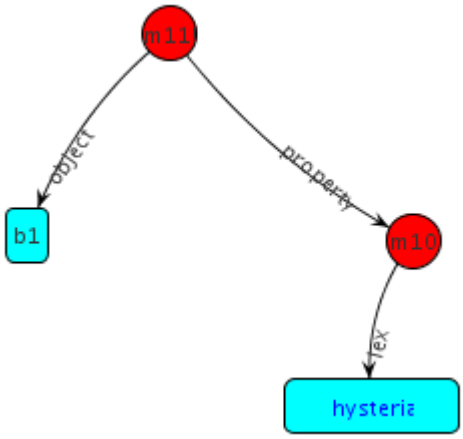
For all x, if x is trembling then x is fearful and unstable.



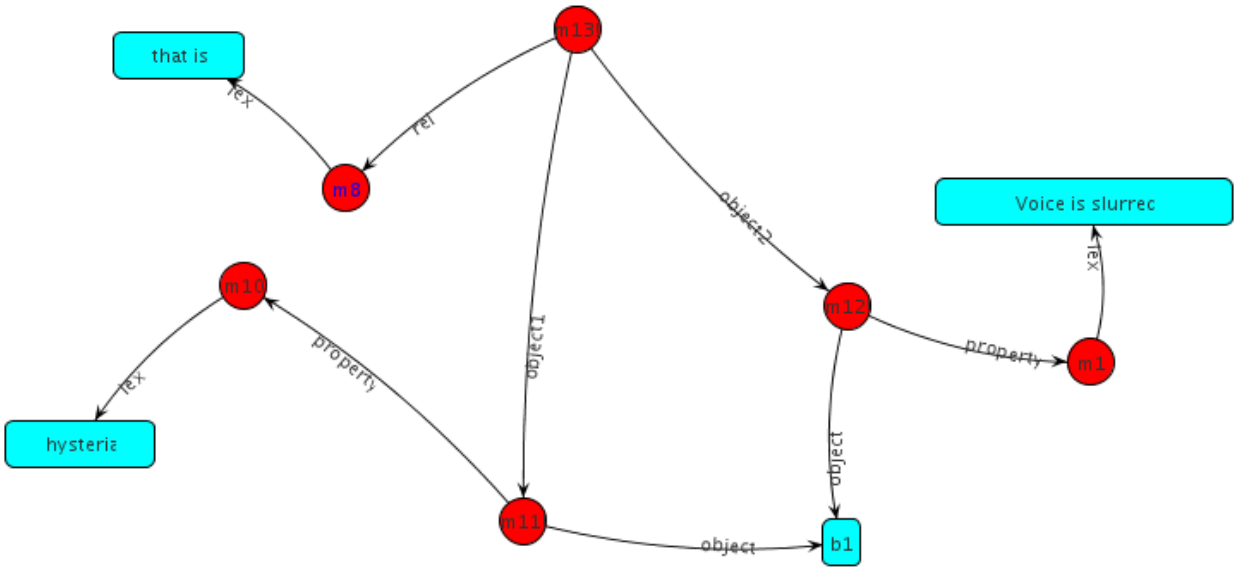
That is.



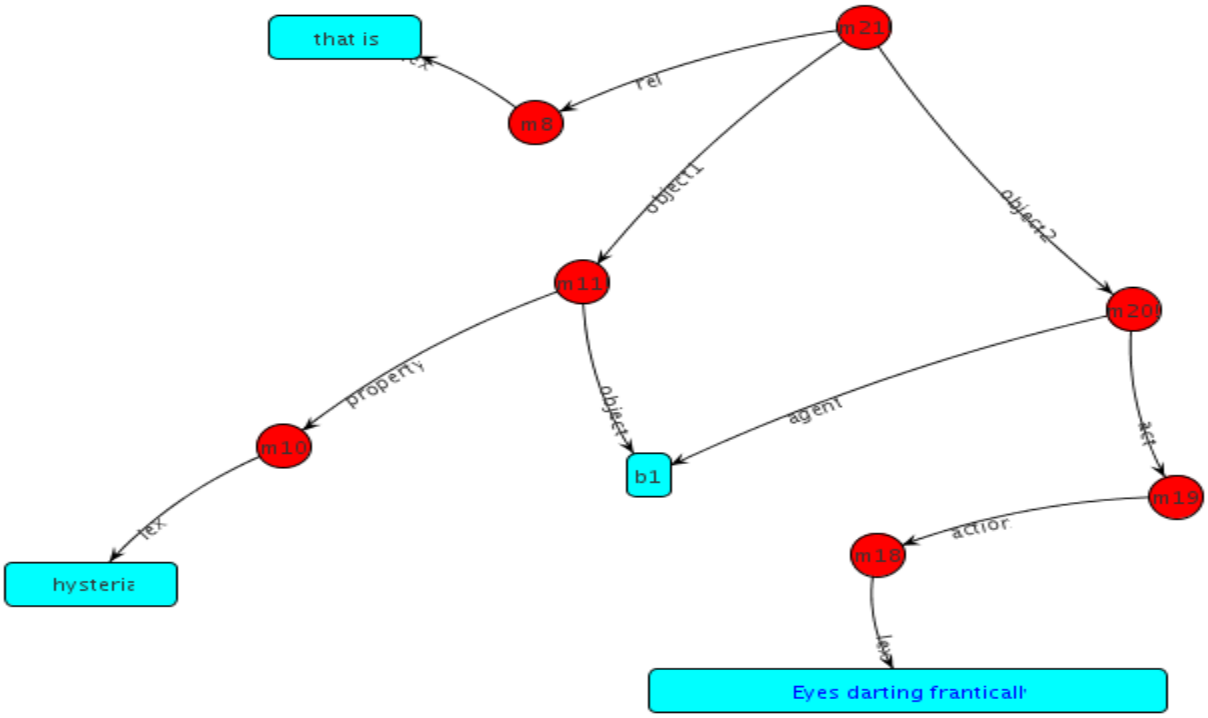
She was close to hysteria.



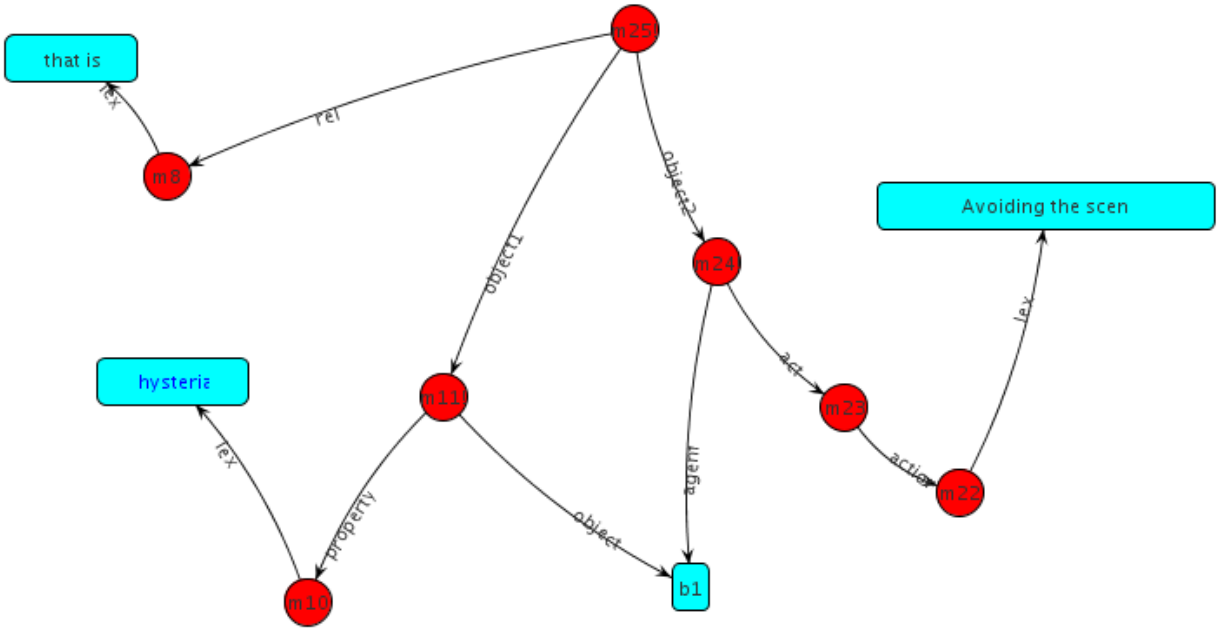
She was close to hysteria that is; her voice was slurred.



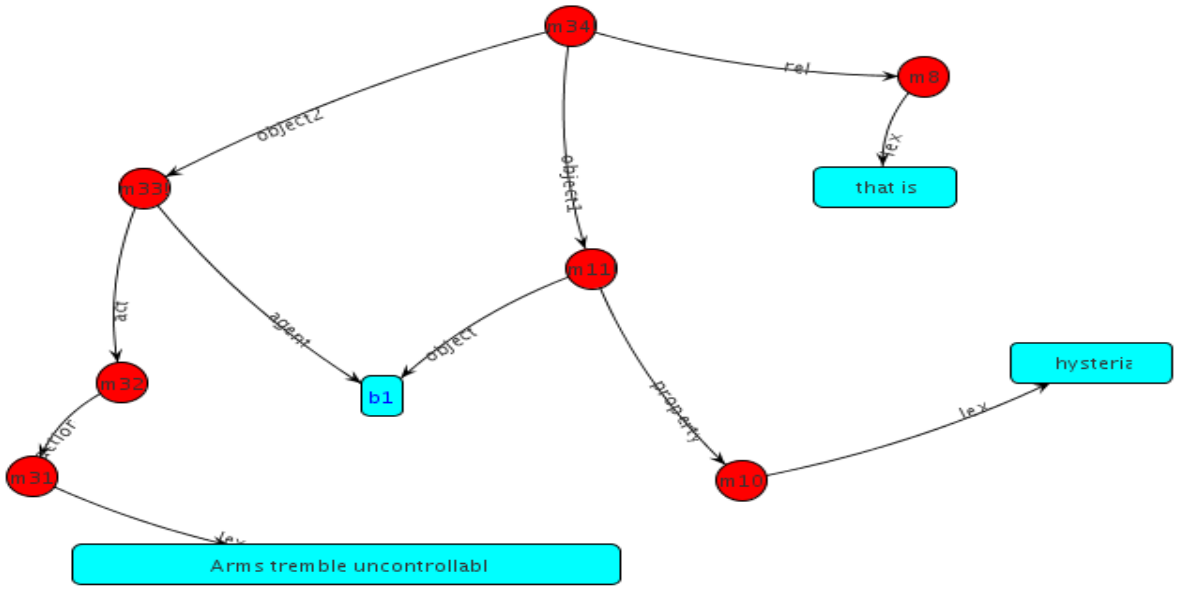
She was close to hysteria that is; her eyes darted back and forth.



She was close to hysteria that is; she was avoiding the scene.



She was close to hysteria that is; her arms and hands trembled uncontrollably.



References

Dulin, Kenneth L. (1970), "Using Context Clues in Word Recognition and Comprehension", *The Reading Teacher* 23(5):440-445, 469

Rapaport, William J.; & Kibby, Michael W. (draft, 2008),

"Contextual Vocabulary Acquisition: From Algorithm to Curriculum"