Contextual Aspects in Text Coherence Analysis [1]

Sanda M. Harabagiu

Parallel Knowledge Processing Laboratory
Department of Electrical Engineering-Systems
University of Southern California
Los Angeles, CA 90089-2562
E-mail: harabagi@usc.edu

Dan I. Moldovan

Department of Computer Science and Engineering
Southern Methodist University
Dallas, TX 75275
E-mail: moldovan@seas.smu.edu

Abstract

Text coherence is a difficult problem in natural language processing. A text is considered to be coherent when sentences follow logically one after the other. In this paper we describe a computational method that provides an explanation for the cohrence of a text. By providing such an explanation, one can infer a number of assertions unstated in a text.

A context consisting of concepts that relate closely to the text is formed as a result of the coherence analysis. The context, in turn, plays an important role in filtering coherence paths as more sentences are analysed.

Our computational method is based on a parallel marker-propagation algorithm that is independent of the size of the knowledge base. We assume that the sentences have been already syntactically parsed and relations between verbs and their grammatical cases were established. At the heart of our system there is a semantic knowledge base consisting of an ontology operating on top of the semantic concept hierarchy provided by WordNet.

Content Area: Natural Language Processing, Contextual Knowledge, Coherence Relations, Marker Propagation.

[1]This research has been funded by the National Science Foundation Grant No. MIP 8902426.

1 The basic idea

In this paper we address the issues of text coherence analysis and context, and their interrelationship. Consider a text understanding system that operates on a large linguistic knowledge base with concepts and relations between concepts. A text context can be constructed by finding the concepts in the knowledge base that relate semantically to the concepts recognized in each sentence. The contextual concepts can be found along or in the vicinity of the text coherence paths between the text clauses. Thus, if we are able to identify such coherence paths we can determine a set of contextual concepts. As more sentences are analyzed for coherence, more such paths will be generated. A method for controlling the growth of the context is computing the level of activity of each concept within the context. This is done by attaching a value to a concept that will be updated when new clauses are processed. Concepts that are not mentioned further in the discourse, or are not semantically related to concepts mentioned, will fade away. The activity levels go up and down as concepts become more or less relevant, respectively.

Furthermore, there is an intertwined relationship between the contextual concepts and coherence paths. While coherence paths generate context, context provides the basis for filtering irrelevant coherence paths.

In this paper we describe a natural language system that uses a knowledge base derived from WordNet, the on-line dictionary developed at Princeton[FGM91]. We designed an algorithm for finding coherence paths using the relations offered by WordNet. The computational mechanism uses a marker propagation paradigm that is highly parallel and asynchronous. Unlike other attempts that use marker-passing [Qui66] [Cha86] [Nor87] [Nor87], we use propagation rules that guide the movement of markers along some selected relations [MoHa55]. The number of coherence paths is significantly reduced by the fact that we use case constraints that need to be satisfied along these paths. Through an example we demonstrate the ideas presented above.

2 The problem of text coherence

A text is considered to be coherent when it is logically connected and intelligible. The logical connections are semantic relationships between concepts. Some of the relations between clauses are elaboration, circumstance, background, justification, causation, etc. Text coherence has been addressed by [Alt85], [Hob85], [Hov93], [MTh88] and others.

In this paper we consider only one aspect of the coherence problem, namely the relationship between clauses within a discourse segment. In particular, we provide a computational method for finding relationships between pairs of clauses and, moreover, explain why such coherence relationships exist. Our solution is to find a semantic path in a knowledge base that links the events or processes in a pair of clauses. The concepts along the path provide an explanation for the relationship between the two clauses.

Consider the text:

```
C1 John hit the ball with a bat.
C2 It landed far away.
```

In sentence `C1`, the verb `hit` has the following several senses: `to knock`, `to give a blow`, `to strike`, `to affect strongly`, `to find accidentally`, `to reach`, `to score`, `to shoot`, `to cause to move by striking`. However, when sentence `C2` follows sentence `C1` we take the `knock` meaning of `hit`. This is because we know that an object may land in some place only if it has been flying, thus there is an entailment between the two concepts. Flying is a way of moving which connects with the `knock` meaning of hit. Thus, to figure out a relational path between `hit` and `land` we have to carry out an analysis far beyond the immediate knowledge base surrounding the word in the input text, and be able to "read between lines."

Humans easily visualize how a ball hit by a bat is first touched by the bat, travels through the air, and finally touches the ground far away from John.

Finding an explanation for the coherence relation is at the heart of natural language understanding since it impacts central problems such as word disambiguation, inference, and others.

3 Finding coherence paths

Below we describe the knowledge base of a system that we designed for concept coherence evaluation, followed by an algorithm based on marker propagation that identifies relations paths between concepts. This algorithm is massively parallel.

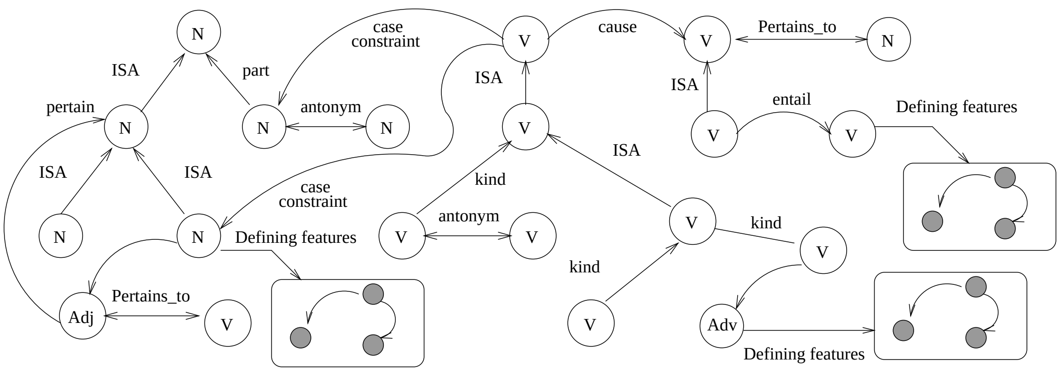

Figure 1: Semantic relations in extended WordNet

Knowledge Base

At the heart of the system is a large linguistic knowledge base, which is build on top of WordNet1.5 developed at Princeton. WordNet, created as a computer based dictionary, is build as a semantic knowledge base covering the vast majority of nouns, verbs, adjectives and adverbs of the English language. The words in WordNet are organized in synsets, which are representations of concepts expressed by synonymous words. The relations from WordNet are the building blocks of our solution for semantic paths finding. The semantic relations between concepts are : hypernymy/hyponymy (ISA relations), meronymy/holonymy (part-of relations), antonymy, troponymy (kind of relations), causation, entailment, and pertanymy

(Pertain relations). We add to these several other relations: (a) case constraint relations which link verb concepts to the most general concepts which may function as case roles (agent, object, instrument, experiencer, etc) for the process described by the verb; (b) defining feature that specifies the properties of concepts; (c) pertain relations based on common morphological properties. Figure 1 illustrates a number of these relations.

Many semantic relations of WordNet are useful for inference because deductions can be done on them due to their transitivity. However, inferences sometimes need to go against these relations and conclusions are not precise. In these cases we perform abductions. However, the semantic relations of WordNet offer a consistent means for inference, allowing their use at different levels of processing. Next we describe the algorithm followed by results.

Algorithm

```
ALGORITHM  CONTROLLER_Find_semantic_path (Clause(i), Clause(i+1))

1. /*  Marker Initialization            */
    for ( cl = 0; cl < 2; cl++){
      for ( j= 0; j < NR_case_roles (Clause(i+cl)); j++){
        1.1.  Create ( Marker 1+cl );
        1.2. Marker 1+cl ->process = get_Concept (Action, Clause(i+cl));
        1.3. if (case(j) == Action){
        1.4.     for(k=0; k<NR_case_roles(Clause(i+cl)) &&( k <> j); k++){
        1.5.          Allocate_memory(new_role);
        1.6.          new_role->case = case (k);
        1.7.          new_role->value = get_Concept(case(k), Clause(i+cl));
        1.8.          Marker 1+cl -> other_roles = Insert (Marker 1+cl ->other_roles, new_role); }}
        1.9. else{ Marker 1+cl ->case = case(j);
        1.10.         Marker 1+cl -> case_value = get_Concept( case(j), Clause(i+cl));}
        1.11. Marker 1+cl -> Path_Length = 0;
        1.12. Marker 1+cl -> Path[0] = get_Concept(case(j), Clause(i+cl));;
        1.13. if (cl == 0) Marker 1 -> other_marker = Marker2;
        1.14. else Marker 2 -> other_marker = Marker1;
        1.15.Marker 1+cl -> Propagation_rule = path_rule;}}

 2. /* Propagation takes place controlled by propagation rules and the processing that takes place in nodes  */
      wait_for_end_of_propagation();
 3. /* Results are collected */
      3.1.  Collect (Collision_Marker);
      3.2.  Eliminate_cycles (Collision_Marker->Path);
      3.3.  Output (Collision_Marker->Path);
```

Figure 2: The controller algorithm for coherence paths exploration

The semantic paths exploration algorithm uses the Marker Propagation Networks [MoHa55] paradigm. It is an MIMD processing, where the execution is controlled by algorithms distributed in the controller code, node processes, and propagation rules carried by markers. The controller algorithm, illustrated in Figure 2, creates and initially places the markers in the network which implements the knowledge base. The control is then passed to the nodes which asynchronously process the incoming markers, following the execution model shown in Figure 4. When the propagation terminates, the controller collects the markers which result at collision and provides the resulting coherence paths.

In the controller algorithm, the marker initialization part is intended to place markers on the concepts of the two clauses under consideration. `Marker 1` is placed in the knowledge

```
Prop_rule Path_Rule (Marker)

Propagate (Marker, (rule1, DEF_FEAT) || (rule1, ISA_V || ENTAIL || CAUSE ||r_ENTAIL || PERTAINS)||
              || (rule2, (ISA_N, PART, PERTAIN), || (rule2, synt-constr));
```

```
Prop_rule rule1   (Marker,  LINK)
Propagate(Marker, LINK);
Marker->Path [++Marker->Length]=Def_Feat;
Marker->Path [++Marker->Length]=this_node;
if( verb (this_node)) {
      Create(Marker_Check_constraints);
      Copy_Fields( Marker, Marker_Check_constraints);
      Marker_Check_constraints->Prop_rule - Check_constr_rule;
      cond ( Marker_Check_constraints);
      if (Present_in_node (this_node, Marker->other_marker)){
         Create (Collision_Marker);
         Collision_Marker->Path = concatenate (Marker->Path,
                                   Marker->other_marker->Path);};
  Marker->Prop_rule = Path_rule;
```

```
Prop_rule rule2   (Marker,  LINK)
Propagate(Marker, LINK);
Marker->Path [++Marker->Length]=Def_Feat;
Marker->Path [++Marker->Length]=this_node;
Marker->Prop_rule = Path_rule;
```

```
Prop_rule Check_constr_rule  (Marker)
for all Marker->other_case {
   Propagate(Marker, other_roles->case);
   repeat {
      Propagate(Marker , r_ISA);
   until (this_node == other_roles->value);
   if (this_node == other_roles->value);
      Propagate(Marker, address (
           Marker->Path[Marker->Length]]);
```

Figure 3: Propagation rules for for coherence paths exploration

base concepts corresponding to $Clause_i$ and `Marker 2` is placed on concepts corresponding to $Clause_{i+1}$. All markers contain a field that initially holds the name of the process described by a clause. When they propagate to other concept nodes, if the case constraints are satisfied, this field is actualized to the name of the new verb concept. A list of all the other cases and their values is maintained in the markers to ease the constraint checking. A mandatory field is the path, where the trajectory of the marker is memorized. Each marker has also a field for the identification of the opponent marker of the collision: `Marker 1` will always collide with a `Marker 2`. Markers are endowed with a control : the propagation rule, which determines at run-time their propagation and processing. Both `Marker 1` and `Marker 2` have the same propagation rule in this algorithm.

```
Node Processing (incoming_marker)

loop (forever) {

   process Propagation_rule (incoming_marker);
   if (synchronization (other markers))
        if (not_present (this_node , other markers))
          place (waiting_list, incoming_marker);
        else continue process Propagation_rule (incoming_marker);
   continue Propagate (incoming_marker);
```

Figure 4: Node processing for the distributed algorithm of coherence exploration

During their propagation, the markers travel along `Defining Features` relations, or along a set of other relations (including `ISA`, `Part_of`, `Cause_to`, `Caused_by`, `Entail`, `reverse_Entail`, `Pertain`), or to the address of concepts which represent the semantic equivalent of some syntactic constraints of the clauses. In addition, some marker propagations are used to test the satisfaction of the case constraints for each step of the propagations above. Details of marker propagation are specified by the code in Figure 3.

During the propagation, the markers reach different nodes, where their propagation rules are executed. Synchronizations between markers are possible with a special construct (the `cond(Marker)` construct) placed in the propagation rule. At each propagation step, the propagation rule is tuned by the outgoing relations. Each rule provides means for collision and makes parallel search for various coherence paths possible. At each step of the propagation loop, the algorithm checks for the collision of odd and even markers. When collision takes place, the presence of a relational path between clauses is signaled. Markers continue to propagate until all nodes related to the originally marked nodes are visited. The algorithm detects and avoids cycles.

Consider the text:

```
C1 John hit the ball with a bat.
C2 It landed far away.
C3 He couldn't find it anymore.
C4 It is a pity,
C5 because it was a new one.
```

Experiment 1 Find the coherence paths between `C1` and `C2`.
For these sentences, the algorithm found the following paths:

```
        Path 1<->2.1
--------------------------------------------------
hit#1->ISA-V->propel->DEF_FEAT->COLLISION->move#1
move#1<-COLLISION<-ISA-V<-fly#2<-r_ENTAIL<-land#3
--------------------------------------------------
```

This is a semantically correct path and it is illustrated in Figure 5. It shows that when `John` (the agent) hits the `ball` (the object) with `the bat` (the instrument), the object is propelled, which is defined as a kind of movement. The landing of an object is entailed by its flying, which is also a form of movement. The case constraints were checked at every step of the path. The case values of `hit#1` hold at `propel` due to the `ISA` relation. The defining feature of `propel` is to `move#1 forward`. Concept `move#1` has an *experiencer* case which takes the value the *object* of `propel`. This is because the `ball` is experiencing a move forward. The *experiencer* case of `move#1` also holds at `fly#3` because of the `ISA` connection and furthermore it holds for `land#3` because of the `Entail` relation. A byproduct of finding the coherence path is the resolution of `it` from `C2` to the noun `ball`.

Only two propagation cycles were necessary to find this path, with 10 marker types for propagation and 12 marker types for case constraints checking. A number of 74 nodes were marked, with a median of 7 nodes per propagation and 9 markers per marker type.

An important source of parallelism in this example is the simultaneous verification of the senses of `hit`, `ball`, `bat` and `land`, which are the starting points of the marker chains. The algorithm also found the following paths.

```
        Path 1<->2.2
----------------------------------------------------------------------
ball#8->DEF_FEAT->throw#5->ISA-V->propel->DEF_FEAT->COLLISION->move#1
move#1<-COLLISION<-ISA-V<-fly#2<-r_ENTAIL<-land#3
----------------------------------------------------------------------
```

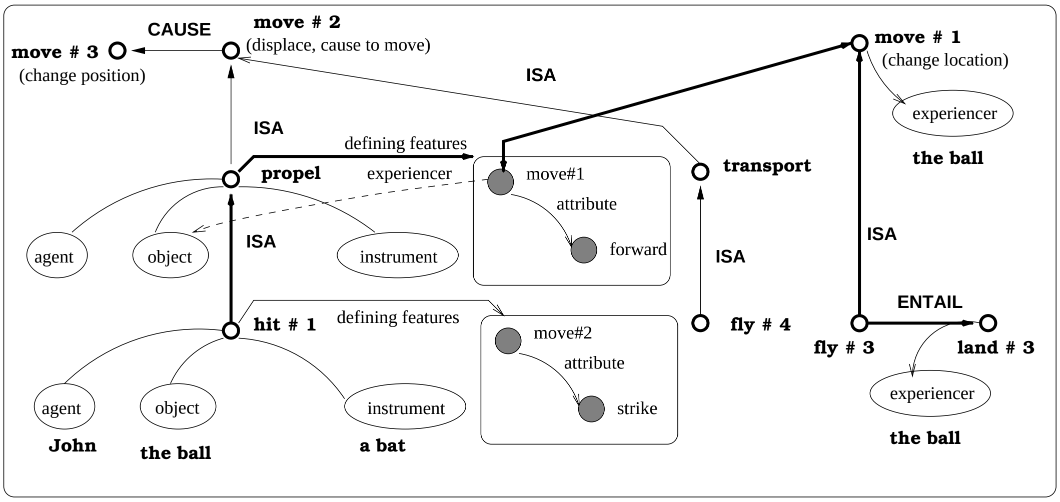

Figure 5: Coherence path between C1 and C2

```
        Path 1<->2.3
----------------------------------------------------------------------
bat#3->DEF_FEAT->hit#1->ISA-V->propel->DEF_FEAT->COLLISION->move#1
move#1<-COLLISION<-ISA-V<-fly#2<-r_ENTAIL<-land#3
----------------------------------------------------------------------

        Path 1<->2.4
----------------------------------------------------------------------
ball#8->DEF_FEAT->hit#4->ISA-V->propel->DEF_FEAT->COLLISION->move#1
move#1<-COLLISION<-ISA-V<-fly#2<-r_ENTAIL<-land#3
----------------------------------------------------------------------

        Path 1<->2.5
----------------------------------------------------------------------
bat#3->ISA-N->club#6->ISA-N->stick#4->DEF_FEAT->wood#1->PART->lumber->
lumber->DEF_FEAT->cut#8->ISA-V->separate#8->ISA-V->COLLISION->move#1
move#1<-COLLISION<-ISA-V<-fly#2<-r_ENTAIL<-land#3
----------------------------------------------------------------------

        Path 1<->2.6
----------------------------------------------------------------------
John->ISA-N->human#2->ISA-N->life form->Pertains_to->shape#7->DEF_FEAT->
arrangement#5->ISA-N->structure#1->ISA-N->cognition->DEF_FEAT->learning#2
->Pertains_to->learn#1->DEF_FEAT->commit#2->ISA-V->pass#10->ISA-V->
transfer#2->ISA-V->COLLISION->move#1
move#1<-COLLISION<-ISA-V<-fly#2<-r_ENTAIL<-land#3
----------------------------------------------------------------------

        Path 1<->2.7
----------------------------------------------------------------------
bat#3->DEF_FEAT->game#3->ISA-N->contest->ISA-N->social event->DEF_FEAT->
person#2->ISA-N->life form->Pertains_to->shape#7->DEF_FEAT->arrangement#5
->ISA-N->structure#1->ISA-N->cognition->DEF_FEAT->learning#2->Pertains_to
```

```
->learn#1->DEF_FEAT->commit#2->ISA-V->pass#10->ISA-V->transfer#2->ISA-V->
COLLISION->move#1<-COLLISION<-ISA-V<-fly#2<-r_ENTAIL<-land#3
-----------------------------------------------------------------
```

The paths Path1-2.2 through Path1-2.4 are such that the semantics of the case constraints are satisfied, whereas paths Path1-2.5 through Path1-2.7 do not satisfy the case constraints. As it can be seen, because of this, the first four paths make sense, while the last three do not.

Experiment 2 Find the coherence paths between C2 and C3.
For these sentences, the algorithm found 12 paths. Below we show the most appropriate semantic paths:

```
        Path 2<->3.1
-----------------------------------------------------------------
find#8->r_ENTAIL->search#1->DEF_FEAT->(try,locate)->ENTAIL->COLLISION->
(reach#2,location)<-COLLISION<-DEF_FEAT<-arrive#1<-ISA-V<-land#3
-----------------------------------------------------------------
        Path 2<->3.2
-----------------------------------------------------------------
find#8->r_ENTAIL->search#1->DEF_FEAT->(try,locate)->ENTAIL->COLLISION->
(reach#2,location)<-COLLISION<-Pertains_to<-location<-Pertains_to<-far
away<-describe<-land#3
-----------------------------------------------------------------
```

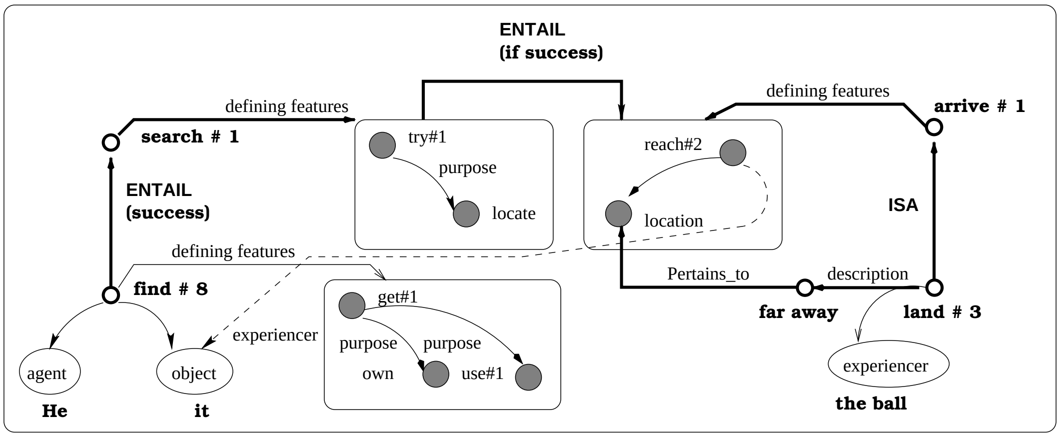

Figure 6: Coherence path between C2 and C3

These two paths have the same same sub-path from `find#8` to the collision, but differ from the collision to `land#3` as can be seen in Figure 6. Path2-3.1 shows that the `ball` landed, thus `arrived`. The defining feature of `arrive` is to reach a location, and this is where the collision with the other path takes place. The other half of the path shows that `he` (the agent) `finds it` (the object), meaning that the agent `searched` for the object, which has the defining feature to `try` to `locate`, which in turn, when successful, `reaches` the object. Here, the reference resolution for `it` in `C3` is resolved by unifying the *object* case for `find#8` to the *experiencer* case of `reach#2`.

Experiment 3 Find the coherence paths between `C1` and `C3`.
For these sentences, the algorithm found 7 paths. Below we show a semantic path which is also illustrated in Figure 7:

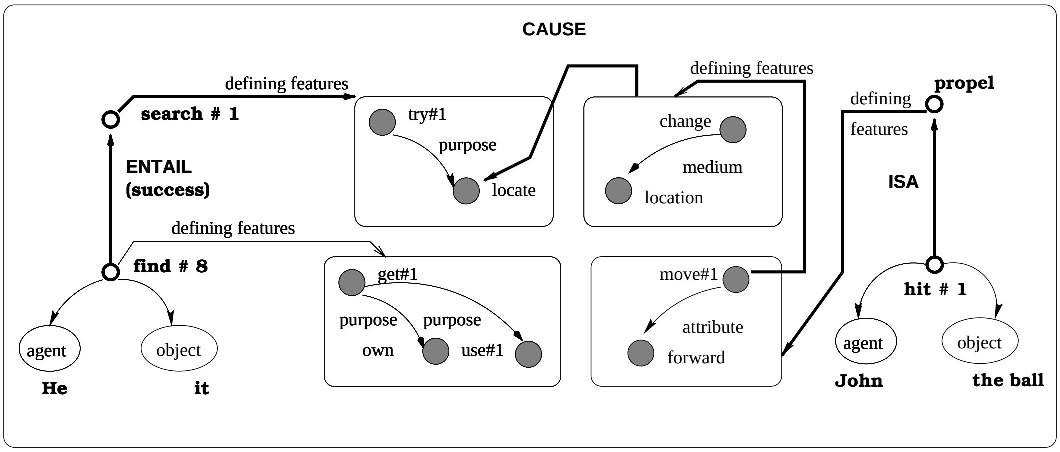

Figure 7: Coherence path between C1 and C3

```
        Path 1<->3.1
----------------------------------------------------------------------
find#8->r_ENTAIL->search#1->DEF_FEAT->COLLISION->(try,locate)<-COLLISION<-
CAUSE<-(change,location)<-DEF_FEAT<-move#1<-DEF_FEAT<-propel<-ISA-V<-hit#1
----------------------------------------------------------------------
```

This path was found because `find#8` entails `search#1` which has as defining feature to try to locate, and this is where the collision took place. The other half of the path starts from `hit#1`, which is a `propel`, whose defining feature is `move forward`. Furthermore, `move#1` has `change location` as defining feature, which is linked to `locate` via the Cause relation.

Experiment 4 Find the coherence paths between `C3` and `C4`.
For these sentences, the algorithm found 4 paths. Below we show a semantic path which is also illustrated in Figure 8:

```
        Path 3<->4.1
----------------------------------------------------------------------
find#8->synt_constr->(negative,modal)->sem_equiv->fail#2->CAUSE->COLLISION
(development,unfortunate)<-COLLISION<-DEF_FEAT-<pity#1
----------------------------------------------------------------------
```

This path was found with the help of the semantic equivalence between not being able to perform an action, e.g., finding, and failing; fail in turn, implies an unfortunate development of that action. On the other hand, the defining feature of `pity#1` is an unfortunate development, where the collision takes place. It this path, `it` from `C4`, which is the *object* of `pity`, refers to an entire action of finding, thus it is a pro-sentential reference for `C3`.

A summary of the coherence paths between clauses of the text is illustrated in Figure 9.

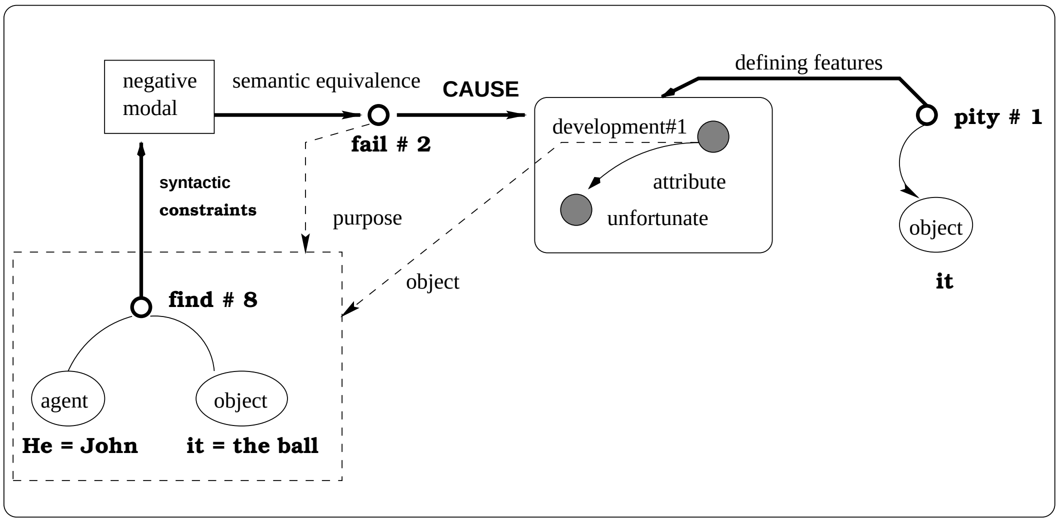

Figure 8: Coherence path between C3 and C4

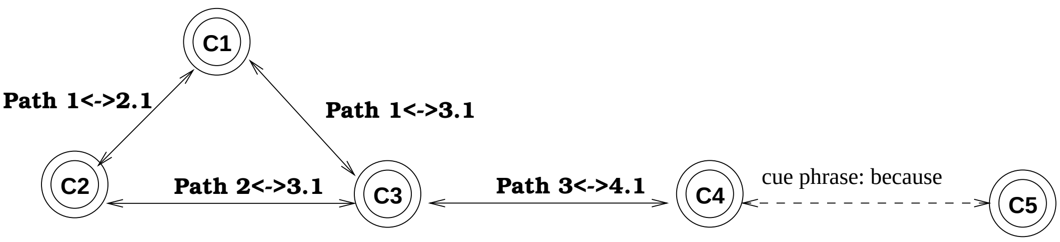

Figure 9: Coherence paths spanning the text

4 Role of context

In this section we will show how a context may be constructed from the coherence paths and, furthermore, how the context may be used to filter out some of the semantically incorrect paths. Table 1 shows the inferences derived from the coherence paths. In addition, we compute some activity levels for the concepts of the discourse. With each occurrence of a concept in the sequence of inference, the activity level of that concept is incremented. Thus we see that the first inference derived from `C1` is `John propelled the ball with the bat`. The activity levels for `John`, `ball` and `bat` are incremented. The next inference, `John moved the ball with the bat` also increments the activity levels for the three concepts. However, the next inference, `It moved far away` only increments the activity for `ball`, due to the resolution of `it` to `ball`. The activity level of `ball` continues to grow. The final result is that concept `John` has activity level of 8, `ball` an activity level of 14, and `bat` an activity level 3. Clearly, `ball` is the dominating concept, as shown by its largest activity.

	Inference sequence derived from text	Activity levels		
		John	ball	bat
1-2	John hit the ball with a bat	1	1	1
	John propelled the ball with a bat	2	2	2
	John moved the ball with a bat	3	3	3
	It moved far away	3	4	3
	It flew far away	3	5	3
	It landed far away	3	6	3
2-3	It landed far away	3	7	3
	It reached location far away	3	8	3
	He tried to locate it	4	9	3
	He searched for it	5	10	3
	He couldn't find it	6	11	3
3-4	He couldn't find it	6	11	3
	He failed to find it	7	12	3
	I is unfortunate that he failed to find it	8	13	3
	It is a pity	9	14	3
4-5	It is a pity	9	14	3
	because it was a new one	9	15	3

Table 1: Inference sequence and concept activity levels

Consider now another text:

```
C1 John hit the ball with a bat.
C6 He felt relieved.
C7 He found peace again.
```

In this example, the focus is on `John`, because `John` is the most active concept. Note that if we try to identify a path between `C1` and `C7` that links `hit` and `found`, the path in Figure 7 will not be valid anymore because the case unification is no longer possible. The object of `hit` is different than the object of `found`. The most active concept is not `ball` anymore, but `John`. Text `C1`, `C6`, `C7` is about `John`, not another object, and this contributes to the invalidation of some erroneous paths found in the knowledge base.

5 Concluding remarks

Inference is a central problem in many AI applications. In this paper we find inferences in the context of text coherence analysis. We have identified a technique for finding coherence paths between concepts in a knowledge base build around WordNet. WordNet is used more and more within the natural language community because of its size and structure. We consider that the development of inference algorithms on the WordNet knowledge base is an important and useful innovation. A feature of the work presented in this paper is that we apply marker propagation techniques to find coherence paths. We do this by guiding the marker propagations in order to control the number of marked concepts. Further elimination of possible paths in large knowledge bases may be achieved by using case constrain satisfaction. In this paper we have shown how the case constrains propagate and change along the relational paths of the knowledge base, but more work is needed to better grasp this idea.

A context results from the inferences provided by the coherence analysis. In order to pinpoint the most specific concepts of the context, activity levels may be computed along the inference chains. We have used a simple scheme of incrementing activity levels whenever a concept occurs, but more elaborate algorithms may be designed for updating activity.

We have shown possible interrelationships between context and coherence paths and came to believe they are complex interrelationships which need to be explored further.

References

[Alt85] R. Alterman. A dictionary based on concept coherence. *Artificial Intelligence*, 25:153–186, 1985.

[Cha86] E. Charniak. A neat theory of marker passing. In *Proceedings of the Fifth National Conference on Artificial Intelligence AAAI-86*, pages 584–588, 1986.

[Fah79] S.E. Fahlman. NETL: A system for representing and using real-world knowledge. MIT Press, Cambridge, MA, U.S.A., 1979.

[Fel90] C. Fellbaum. English verbs as a semantic net. Technical Report CSL 90-43, Princeton University, July 1990.

[FGM91] C. Fellbaum, D. Gross, and G. Miller. Wordnet : A lexical database organized on psycholinguistic principles. In Lawrence Erlbaum Associates, editor, *Lexical Aquisition: Exploiting On-Line Resources to Build a Lexicon*, pages 141–170. Zernik, U., Washington, 1991.

[Gro83] B.J. Grosz. The representation and use of focus in a system for understanding dialogs. In *The Proceedings of the International Joint Conference on Artificial Intelligence, IJCAI-77*, pages 67–76, 1983.

[GS86] B.J. Grosz and C.L. Sidner. Attention, intention and the structure of discourse. *Computational Linguistics*, 12:175–204, 1986.

[Haj87] E. Hajicova. Focusing - a meeting point of linguistics and artificial intelligence. In P. Jourand and V. Sgurev, editors, *Artificial Intelligence: Practice and Experience*, pages 311–321. Elsevier Science Publishers, North-Holland, 1987.

[Hen88] J.A. Hendler. Intergating Marker–Passing and Prolem Solving : A Speading Activation Approach to Improve Choice in Planning. Lawrence Erlbaum Associates, 1988.

[Hob85] J.R. Hobbs. On the coherence and structure of discourse. Technical Report CSLI-85-37, Stanford University, 1985.

[Hov93] E.H. Hovy. Automated discourse generation using discourse structure relations. *Artificial Intelligence*, 63:341–385, 1993.

[MTh88] W.C.. Mann and S.A. Thompson. Rhetorical Structure Theory: Toward a functional theory of text organization. *Text*, 8-3:219–281, 1988.

[MoHa55] D. Moldovan and S. Harabagiu. Marker Propagation Networks. Technical Report 95-CSE-18, Southern Methodist Univbersity, 1995.

[Nor87] P. Norvig. Inference in text understanding. In *National Conference on Artificial Intelligence AAAI-87*, pages 561–565, 1987.

[Qui66] M.R. Quillian. *Semantic Memory*. PhD Dissertation, Carnegie Institute of Technology (Carnegie Mellon University), Pittsburg, PA, 1966.

[Sid84] C.L. Sidner. Focusing in the comprehension of definite anaphora. In M. Brady and R.C. Berwick, editors, *Computational Models of Discourse*, pages 267–330. The MIT Press, 1984.