Discourse planning as a transformational process

Jeremy Dohmann

May 7, 2018

Abstract

This paper proposes a theory of discourse planning as a transformational process. I postulate that discourse planning consists of four primary types of knowledge:

- Content-knowledge; this is information relevant to the topic of discourse or knowledge from the world
- Relational knowledge; this is knowledge about how to derive the relationships between elements in the universe, it permits the establishment of logical relationships including: cause/effect, topical relation, correlation, etc.
- Transformational knowledge; this is, potentially language agnostic, information about how to transform some units of knowledge into more complex units of knowledge, given the structure of the subelements and (optionally) semantic predicates relating the pieces of knowledge; examples include passivization, which merely changes the presentation of knowledge, nominalization which does likewise, coordination which syntactically realizes certain types of relational knowledge, etc.

- Procedural knowledge; This is knowledge which relates the above three types of knowledge to the goals of an agent, resulting in a structure appropriate for accomplishing the goals; this is essentially the "planning" aspect of the system; an example would be: the agent would like to make a case for public service playing a larger role at an institution dominated by financial interests, the agent would then use the content, relational, and transformational knowledge to select appropriate transformations in an order that creates a rhetorical structure appropriate for accomplishing this goal

I will then explore the atomic units of each of those types of knowledge and present a theory of representation for those atomic units.

Finally, I will briefly outline a theory for how these units interact, positing that the process is fundamentally built on transformation (i.e., functional) application informed by the first three types of knowledge above and guided by the last form of knowledge.

I will then relate this model to Ross' Performative Sentence Hypothesis by decomposing clauses into 3 types: primary content clauses, metacommentary subordinate clauses, and rhetorical subordinate clauses. I will explore several sentence-level examples of this typology taken from Ross (1970) and Rutherford (1970) and another discourse-level example taken from The Harvard Crimson (2018).

1 Knowledge and representation

1.1 Content-based knowledge

Content-based knowledge is explicit knowledge about the existence of entities in the universe, literal relationships between them, and descriptions of those entities. This can be information specifically relevant to the topic of the discourse, or it can be information known by the agent about the universe.

In terms of content, these can look essentially like dictionary entries, or thesaurus entries, or even things which are unverifiable, or seem like opinions.

Examples, in natural language form, are given below

> "Dogs have four legs", "Dogs are furry", "Ann can swim", "Swimming is good", "Some people think swimming is good", "Rain is caused by clouds", "There is a genetic component to language", "Noam Chomsky believes there is a genetic component to language", etc.

The important thing to note here is that, because it can include logical relationships, opinions, generalizations about what things may be true for stereotypical elements of a class, or generalizations about what others may think or believe, content-based knowledge really encodes the "beliefs" an agent holds.

Content-based knowledge is the meat of discourse; a discourse agent accomplishes its goals only through reference to content-based knowledge. If that knowledge is shared with the reader it can gainfully be used as the basis of relational knowledge, or argumentation. If that knowledge is topic-specific its use can be primarily that it is previously unknown to the reader. Thus relational knowledge can proffer new forms of argumentation, but only by reasoning over what the agent knows. Procedural knowledge can accomplish the goal of convincing, informing, or entertaining a reader, but only through reference to content and the relationships the user can derive from it.

There are many different ways to represent knowledge. At the very least, our desire to encode generalizations require us to use a representation system at least as expressive as first-order logic. Our desire, explored in the next subsection, to permit (possibly erroneous, c.f. our argument that these are more "beliefs" than knowledge) reasoning, deduction, and inference over knowledge suggests that there needs to be a certain fuzziness to our beliefs. Finally, since this is fundamentally a computational theory masquerading as a linguistic theory masquerading as a cognitive theory, we ought to use an approach that, minimally, is amenable to computational forms of reasoning

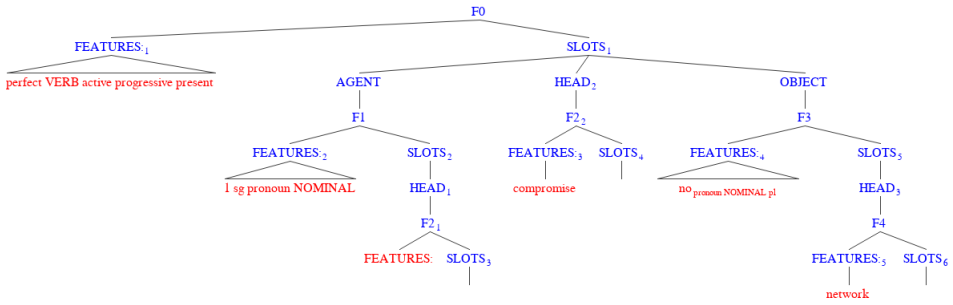
and has some sort of psycholinguistic basis.

I typically like to leave myself unencumbered by lambdas so I'm a big fan of Walter Kintsch's Construction Integration Model, a form of "propositional" logic similar to the way in which propositions are stated in Prolog. It is fundamentally a cognitive model, but it has been extended by Kintsch's group at UC Boulder to have computational implications as well. In Kintsch's model, sentences encode logic primarily in semantic tree-form, though through a complex of inference relations sentences can possibly represent generalized directed graphs of knowledge.

I think the dangers of this is that it creates an unrestricted semantics (not to get into theory of computation, but our reasoning is essentially as expressive as a universal model of computation) which doesn't accord with the fact that our syntax is compositional in a very simple way, and is in fact a *restricted* computational model (context free, or mildly context-sensitive depending on who you ask). As a matter of fact, many of the higher order logical relations in Kintch's CIM (the kinds of directed edges that turn his trees into graphs) come about as the result of inferences. I prefer to allow inferences to exist on a separate level of representation, because they more appropriately belong to the realm of reasoning than they do the realm of language and syntax (while it *is* symbolic, its application or derivation is very fuzzy - perhaps even fundamentally connectionist)

Thus I propose a simple tree representation based on Minsky's frame semantics, where trees have slots (labeled edges) leading to embedded frames or leaf frames containing, in our case, textual information. I extend this formalism to include annotations (also called features) appended to frames as well, representing semantic or syntactic information.

An example is given below:



This sentence encodes a meaning identical to "I am compromising networks."

The particular choices of what annotations mean or how to encode syntactic/semantic information in annotations, or even what slot names to use are empirical questions. The fundamental postulate we give here is that we can encode the kind of knowledge defined in this section using abstract trees consisting of labeled edges leading to other trees or lexical information and sets of features/annotations describing whatever semantic/syntactic phenomena we deem relevant to the meaning of a piece of knowledge.

1.2 Relational knowledge

If content-based knowledge stores topic-specific knowledge and general beliefs, relational knowledge encodes how to generate more knowledge/beliefs from that knowledge. This is a well-studied problem with many plausible solutions. Viewed from a distance, everything from the constructionist logic of theorem provers, to the unification procedure of Prolog, to the domain-engineered knowledge of WordNet, to the connectionist reasoning of a neural network all satisfy the requirements of "relational knowledge".

Thus in many ways relational knowledge isn't really knowledge in *itself* but rather knowledge of an algorithm, or a calculus, or a strategem for deriving new insights from the beliefs and knowledge of the last section.

For instance, relational knowledge could include a component based on constructionist logic that maps the content-based knowledge primtives "All

men wear hats" and "John is a man" to "John wears hats".

Relational knowledge could be the unification procedure by which the Prolog knowledge base:

```
parent_of(John, Mary).
parent_of(Charles, John).
related(X,Y) :- parent_of(X,Y).
related(X,Y) :- parent_of(Y,X).
related(X,Y) :- related(X,Z), related(Z,Y), X!= Y.
```

and the query:

```
related(Charles,X)?
```

produces the new knowledge:

```
related(Charles,John). related(Charles, Mary).
```

Relational knowledge could also be encoded in a rule-based production system powered by the Rete algorithm, as in the production language OPS5.

Relational knowledge could also take the form of a connectionist machine learning algorithm that takes in two inputs and returns a prediction about whether there is a causal relationship between them. Or an algorithm which takes inputs and classifies them according to some predefined, or some latent, undiscovered categories.

There are many ways to skin a cat and the sheer volume of ink spilled championing *incredibly* different ways to approach the task that we've here called "relational knowledge" speaks power to the claim made earlier that the "inference" knowledge we describe exists at a far more unrestricted computational level than does regular content-based knowledge.

Thus, though we have not come upon a precise definition of how to represent relational knowledge, we have established the very clear dichotomy that content-based knowledge is beliefs, and relational knowledge is encoded in algorithms and procedures defining how to infer *new* beliefs.

1.3 Transformational knowledge

Transformational knowledge is to procedural knowledge what content-based knowledge is to relational knowledge.

Transformational knowledge defines the "rules of engagement" when it comes to composing beliefs into complex sentences and ordering complex beliefs into a discourse. Transformational knowledge is thus inherently *functional* - it is a set of functions describing the inputs they take and the outputs they produce. They are the elementary operation in discourse planning, they are the *what* and *how* to procedural knowledge's *when* and *where*.

A huge list of example functions is given in appendix A.

A transformation straddles syntax, semantics, and pragmatics. It tells us how rephrasing the description of an item is realized syntactically (ex 1-9 in the appendix) and it tells us how to combine beliefs into more complex beliefs, whether through conjunction, contrast, etc. It thus works on the atomic units of content and relational knowledge. For example, there may be a transformation representing how cause and effect is realized syntactically, which operates on two content-based knowledge units iff relational knowledge states there is a cause and effect relationship between them.

One can say that transformations encode a type of "compiled" language competence. Skills that all language users have telling them how to combine things given certain predicates and knowledge. They are far more constrained than relational knowledge because they accomplish a relatively narrow set of tasks. Though there may be a billion ways for one thing to cause another thing (an inference which is the purview of relational knowledge), there are only a few ways to say "Because x, y" (all of which are identical up to some sort of syntactic isomorphism). Thus the utility of thinking about language in this manner, is that it abstracts away the very complicated task of reasoning, and exploits the high degree of regularity with which we transform broad classes of semantic units into other classes of semantic unit given satisfaction of complex logical predicates. It is this insight that motivated the choice to

represent functions as operating on patterns and predicates, described below.

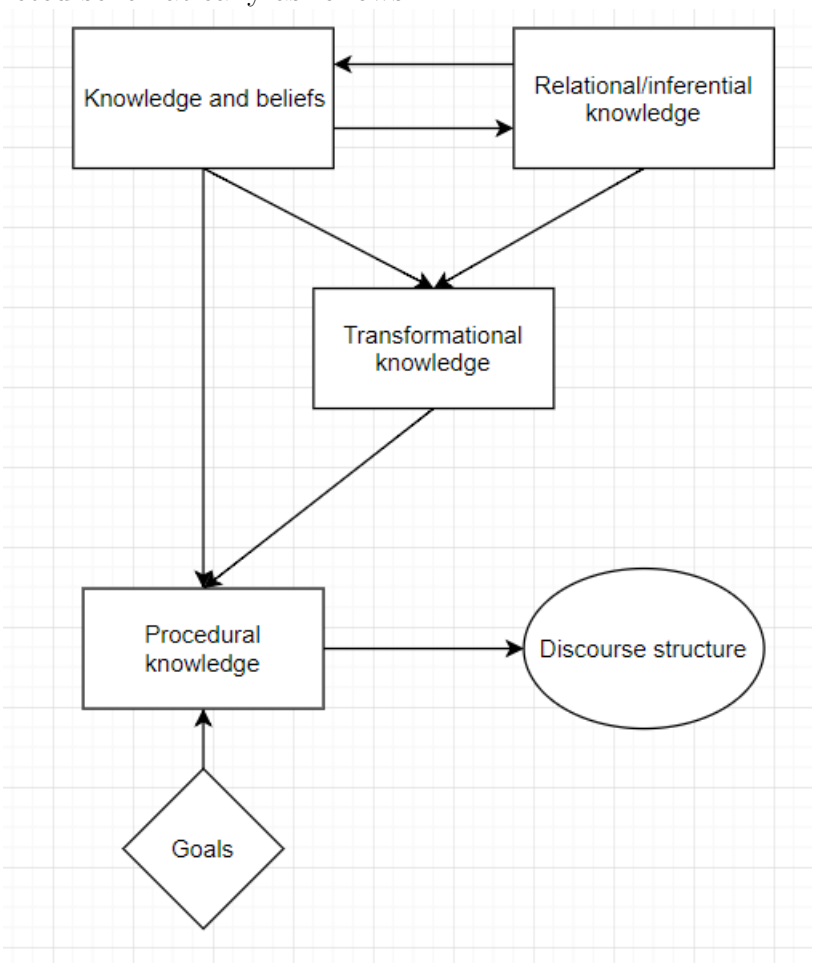
I am currently undertaking work to develop a programming language Taml (Tree abstraction metalanguage) that attempts to define a semantics for writing tree transformations such as those needed in our model of discourse planning. It is also a long-term aim to make Taml incorporate reasoning based on relational knowledge and procedural knowledge, but as of right now its primary task is to encode transformations and the content-knowledge units they act on (given in the tree format described in 1.1). Our design has assumed that transformations are defined in terms of templates of the kinds of inputs they can operate on (these are the *patterns* given in the function definition and operate primarily on content-based knowledge) and predicates under which they are allowed to be applied (these operate on the level of relational knowledge). In the appendix, the "template" is represented above the arrow, indicating patterns for both slots and contents of the inputs, as well as annotations on the RHS. Some examples of predicates (described in plain English) are given in exs 10-12 in appendix A. It should be clear that the transformations exploit regularities by using patterns to abstract over the structure of content-knowledge inputs and by using predicates to abstract over relational knowledge *as applied to* those inputs.

1.4 Procedural knowledge

Procedural knowledge is the least well-defined of our four types of knowledge given. Though it is essentially analogous to relational knowledge insofar as it defined procedures for deriving one thing from another, it is much more complicated. This is so because it operates on all three forms of knowledge given above *and* it also concerns itself with accomplishing *goals*, a concept which is the subject of much research in AI and planning systems. Though various forms of goals have been proposed for many different tasks in AI, it is an interesting and open question what the best way of encoding discourse-based goals is. For more discussion, look into any literature on cognitive

architectures such as SOAR, BDI, and GOMS.

Thus procedural knowledge encodes how the reasoning agent, given knowledge, the ability to generate relational dependencies from that knowledge, and competency regarding how to compose knowledge into more complex units (i.e., transformational knowledge), chooses which functions to apply to what pieces of knowledge, in order to accomplish its goals. This can be depicted schematically as follows:



2 Clause typology

I will now give a brief exposition on how this lends a window into the way discourse works in English.

I will broadly limit myself to clauses of 3 types: primary content clauses, metacommentary subordinate clauses, and rhetorical subordinate clauses.

1. **Primary content clauses:** these are the clauses which form the nuclei of sentences in discourse. They are composed of content-based knowledge, usually aggregated in some complicated way. A fine-grained analysis of how to build complex content-based knowledge sentences from smaller units is beyond the scope of this paper. They are marked in green in my examples.

2. **Rhetorical subordinate clauses:** These are clauses which contain content, but their coordinate with respect to a main clause reflects some sort of rhetorical aspect of the discourse. They often reflect a relationship that the author *believes* or would like to *argue* holds between two pieces of information. They are also typically related to the goal, insofar as their presence is used to suggest points, or subpoints related to the accomplishment of the author's goals. These are highlighted in blue in the examples.

3. **Metacommentary subordinate clauses:** These clauses are the residue of the implicit performative deletion Ross describes in hs Performative Sentence Hypothesis. Metacommentary differs from content because it is self-aware. It concerns itself not with the subject matter, but with establishing coherence by commenting on the *discourse itself*. It thus, cannot be accounted for by using an analysis of discourse where text comes exclusively from content, but rather where text can also come as an epiphenomenon of the content. These are highlighted in yellow in the examples.

My thesis, borne out in the examples below, is that the first type derives comes from basic logical processes by which simple beliefs aggregate to become complex sentences. I will then show the second type derives from applying high order reasoning and transformational rules encapsulating notions such as "imply X causes Y given prior belief Z" to accomplish goals such as "convince audience X causes Y". Finally, I will provide examples that the third type comes from the reasoning agent attempting to enhance coherence and clarity by "explaining" its choice of transformation applications. Thus "metacommentary", rather than coming from deletion from an implicit performative can be thought of as a natural part of the cycle of function application, in which the reasoning agent decides clarity or rhetorical force would be appreciably improved by outputting a metacommentary on why it just stated what it did. This metacommentary is often used to explain why something is being placed in the discourse where it is, but I think it can generally be used as a justification for any given functional application.

A corollary of this analysis, which places primacy on the first type of clause, is that a discourse could function entirely with only clauses of the first type. It is clauses of type two that make the claims novel, interesting, provocative, and engagine, and it is clauses of the third type that make the claims have force and coherence.

3 Examples

3.1 Performative Sentence Hypothesis

All the examples below are from the appendix of Rutherford (1970) unless otherwise noted.

As you can see, in all of the examples below, the true content of the sentence is contained the type-1 clauses, highlighted in green. Though we have not explicitly given attention to the representation of the *reasons* the procedural knowledge chooses a transformations, this analysis shows that

there must be some sort of way to invert the reasoning back into text in order to produce the metacommentary in yellow. Another reflex of this hypothesis, is that we can analyze examples like those given below to peer into the kind of reasoning that may take place in the procedural knowledge.

For example, 82e-f and 84a suggest that procedural knowledge may make decisions based on the structure of the argument. If the topic of discussion has suddenly changed to a *new* topic, we may need to establish coherence by outputting metacommentary such as that in 82e, whereas if the subject changes to a topic discussed earlier, we may signal this by outputting metacommentary such as that in 82f. Finally, 84a shows that if we begin our argument by looking at some subpart of it, we may establish coherence by explicitly stating that we are narrowing our scope temporarily. All of these examples show that *if* metacommentary truly is a reflex of procedural knowledge attempting to justify transformation application, then some of the rationale that goes into transformation application must address things such as the order in which introduce topics and the process by which we narrow or expand our scope in the course of discussion.

78 and 82b look a lot more like pragmatic metacommentary, designed to soften the content. If so, that means the metacommentary here isn't a description of *why* a particular utterance S is being uttered, but rather serves pragmatic considerations (softening blunt statements) within the context of a discourse in which S has been stated. 64 also looks like a pragmatic metacommentary, but instead one which is meant to hedge statements which may be incredulous to the reader - this is a complicated sort of reasoning which must imagine what other agents in the environment believe, or what types of beliefs are inherently credible or incredible.

Finally, I'm not sure exactly what to make of 84e. To me, it feels like filler that can be used in many different contexts, all of which are rhetorical. For instance if S is contrary to something said earlier, we may introduce that refutation by outputting metacommentary "Strictly speaking", thus an

example such as 84e shows that the procedural knowledge keeps track of when one thing may be contrary, or ostensibly contrary to other elements of the discourse.

64) Ann can swim; but if you don't believe me, just watch her (Ross 1970)

78) To tell you the truth, I don't know.

82b) To put it bluntly, S

82e) To change the subject, S

82f) To return to our original subject, S

84a) Looking first at the problem itself, S

84e) Strictly speaking, S

3.2 Discourse-level analysis

At a risk of beating this subject to half to death, I will only speak about the type-2 clauses in the example below and discuss what they may suggest about the rule selection process needed for procedural knowledge.

The first example S1="Situated in a milieu...norm" is an interesting form of prefatory clause that doesn't explicitly do *anything*. It really just causes two ideas to co-occur. This sort of occurrence, in this context, means that the author would like to suggest that S1 is related to S2. In fact, recurrent throughout the op-ed is that Harvard's gospel of wealth squelches public service, and that public service is thus deserving of an op-ed to remind people of its importance. Thus S1, though it is inherently derived from content, must be chosen to be coordinated with S2, via some rule which coordinates two statements if they are plausibly causative with one another. The appropriateness condition then, would be triggered whenever the author has a goal of exposing this causative relationship.

Thus this tells us that the transformational knowledge must be able to take predicates indicating weak causative or correlative relations and that

the procedural knowledge must be able to trigger these transformations if it has a goal of suggesting that in fact that causative relationship does hold.

The same analysis can be given to the second type-2 sentence in blue. Except this time, the transformational rule must be able to relate phenomena which occur at some institution (capitalizing etc. at Harvard) to events or opinions held at that institution (questioning, etc.) and suggest a causative relationship holds between the things which occur at the institution and the attitudes evinced there. At the procedural level, the analysis is essentially the same as that above, except it has the added element that the goal involves showing some X (potentially at Z) causes phenomenon Y at Z.

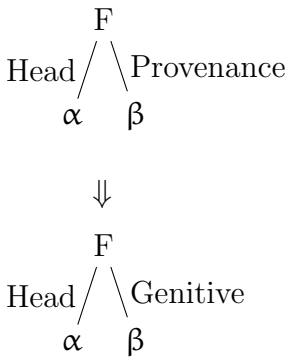
> Yet, in spite of this meaningful work, which occurs both through PBHA and many other outlets at Harvard, the value of public service often remains an afterthought here. Situated in a milieu where finance, consulting, and technology careers are the norm, public service's dearth of monetary incentives on campus exacerbates its perception as a risky career path. Some have even questioned the purpose of service at an institution like Harvard, where it seems like capitalizing on the ostensibly stuffed wallets of students and graduates is the more effective option.
>
> To the contrary, public service offers crucial experiences for students and an enduring impact on our constituents. In particular, PBHA is a rare opportunity for meaningful community engagement, student development, and social justice.

References

- "The Role of Knowledge in Discourse Comprehension: A Construction-Integration Model" by Walter Kintsch (1988)
- "Some Observations concerning Subordinate Clauses in English" by William E Rutherford (1970)
- "On declarative sentences" by John R. Ross (1970)
- "A Different Kind of Harvard House' by Jang H. Lee (2018)

4 Appendix A: Example transformations

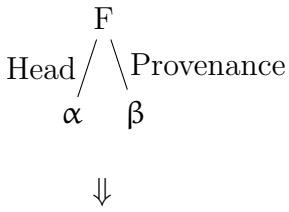
1. Ownership attribution via genitive



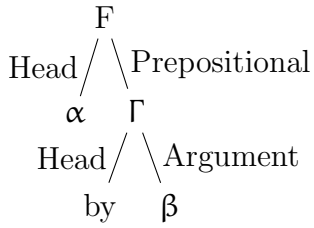
β: [has_agency, NOMINAL]

e.g. "malware from Jeremy" → "Jeremy's malware"

2. Authorship attribution via PP



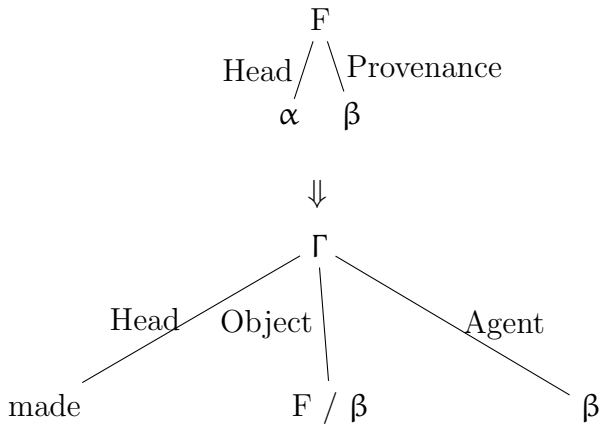
β: [has_agency, NOMINAL]



Γ: [PREPOSITIONAL]

e.g. “malware from Jeremy” → “Malware by Jeremy”

3. Process description via agentive VP

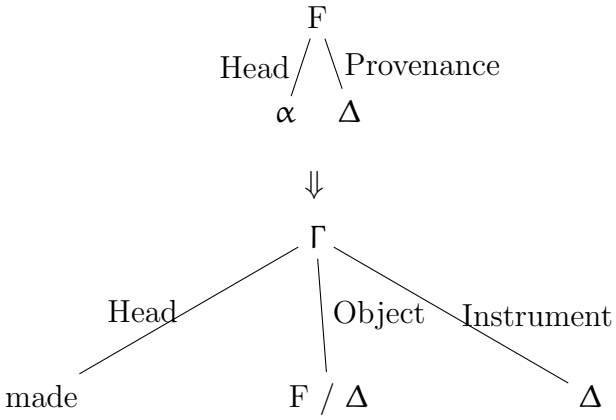


β: [has_agency, NOMINAL]

Γ: [VERB, active]

e.g. “malware from Jeremy” → “Jeremy made malware”

4. Process description via instrumental VP

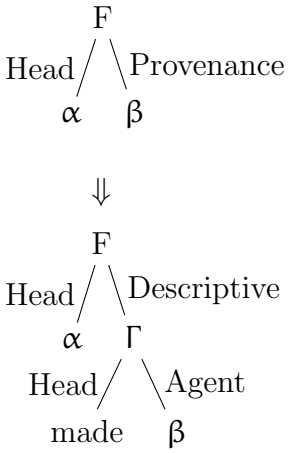


Δ: [no_agency, NOMINAL]

Γ: [VERB, active]

e.g. "malware in the C programming language" → "Malware was made with the C programming language"

5. NP with passive verbal modifier

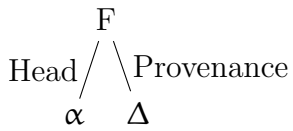


β: [has_agency, NOMINAL]

Γ: [passive, VERB]

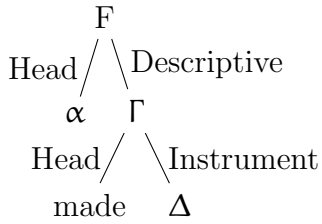
e.g. "malware from Jeremy" → "Malware made by Jeremy"

6. NP with passive verbal modifier showing instrument



Δ: [no_agency, NOMINAL]

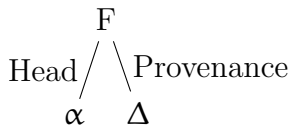
⇓



Γ: [passive, VERB]

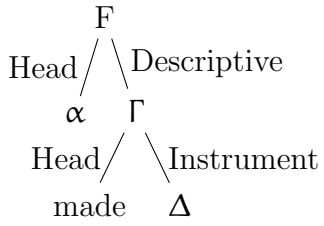
e.g. "malware in the C programming language" → "Malware made with the C programming language"

7. NP with relative clause modifier showing instrument



Δ: [no_agency, NOMINAL]

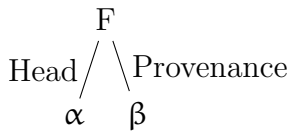
⇓



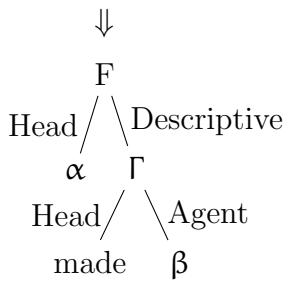
Γ: [passive, VERB, relative]

e.g. "malware in the C programming language" → "Malware which was made with the C programming language"

8. NP with relative clause modifier



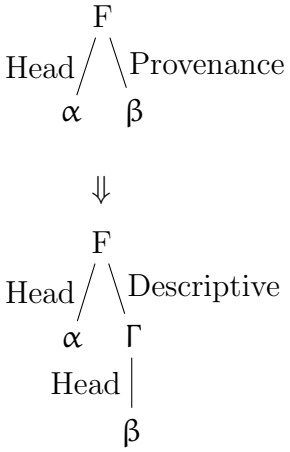
β: [has_agency, NOMINAL]



Γ: [passive, VERB, relative]

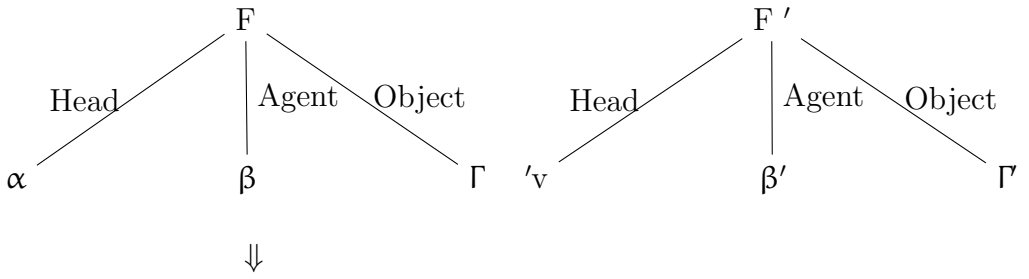
e.g. "malware from Jeremy" → "Malware which was made by Jeremy"

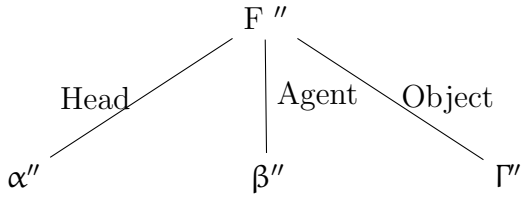
9. NP with provenance as prepended adjective



e.g. "candlestick (from Germany |from a lathe)" → "(German |lathe-made) candlestick"

10. Conjunction with user defined equality predicates





e.g. "The FBI collected samples of malware_i" AND "DC3 collected sampls of malware_i written in C" $\rightarrow$ "The FBI and DC3 collected samples of malware_i written in C."

With equality predicates:

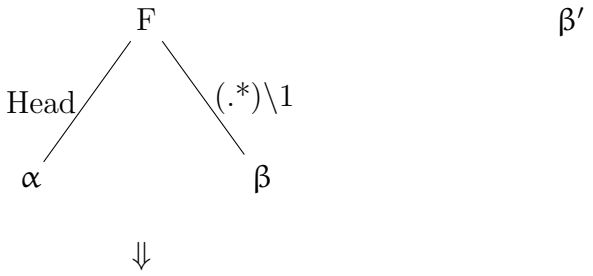
$\alpha = \alpha'$ iff they are synonyms

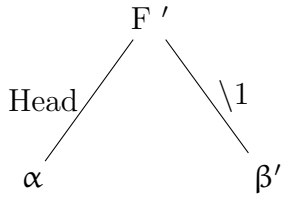
$\Gamma = \Gamma'$ iff their heads are the same referent

Additionally, $\Gamma'' := \Gamma \cup \Gamma'$, defined as the union of their slots and annotations

and $\alpha'' :=$ either α or α' chosen according to some semantic policy

11. Replacing less descriptive object with more descriptive



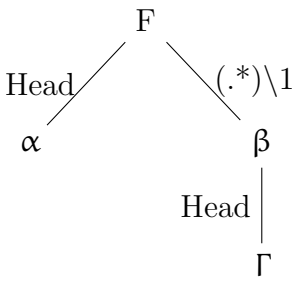


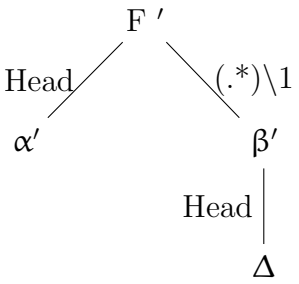
e.g. “The FBI collected samples of malware_i” AND “Malware_i was made with Sakula”→ “The FBI collected samples of Sakula malware_i.”

With predicate:

$\beta \subset \beta'$

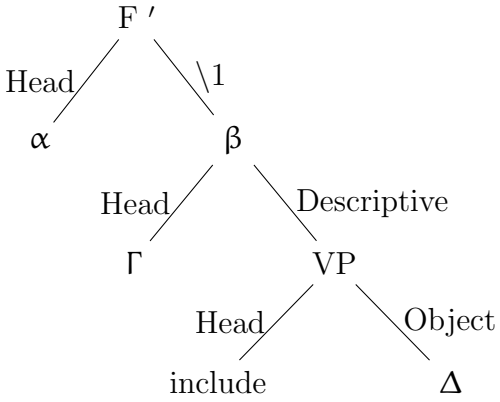
12. Replacing non-head type in verb frame with type + subtype/variant qualification





$\beta, \beta' : [NOUN]$, $F, F' : [VERB]$

⇓



e.g. "The FBI collected samples of malware_i" AND "The FBI collected media_center.exe." AND "media_center.exe is a variant of malware_i."→ "The FBI collected samples of malware_i, including media_center.exe."

With predicates:

Δ variant of Γ

α synonym of α'