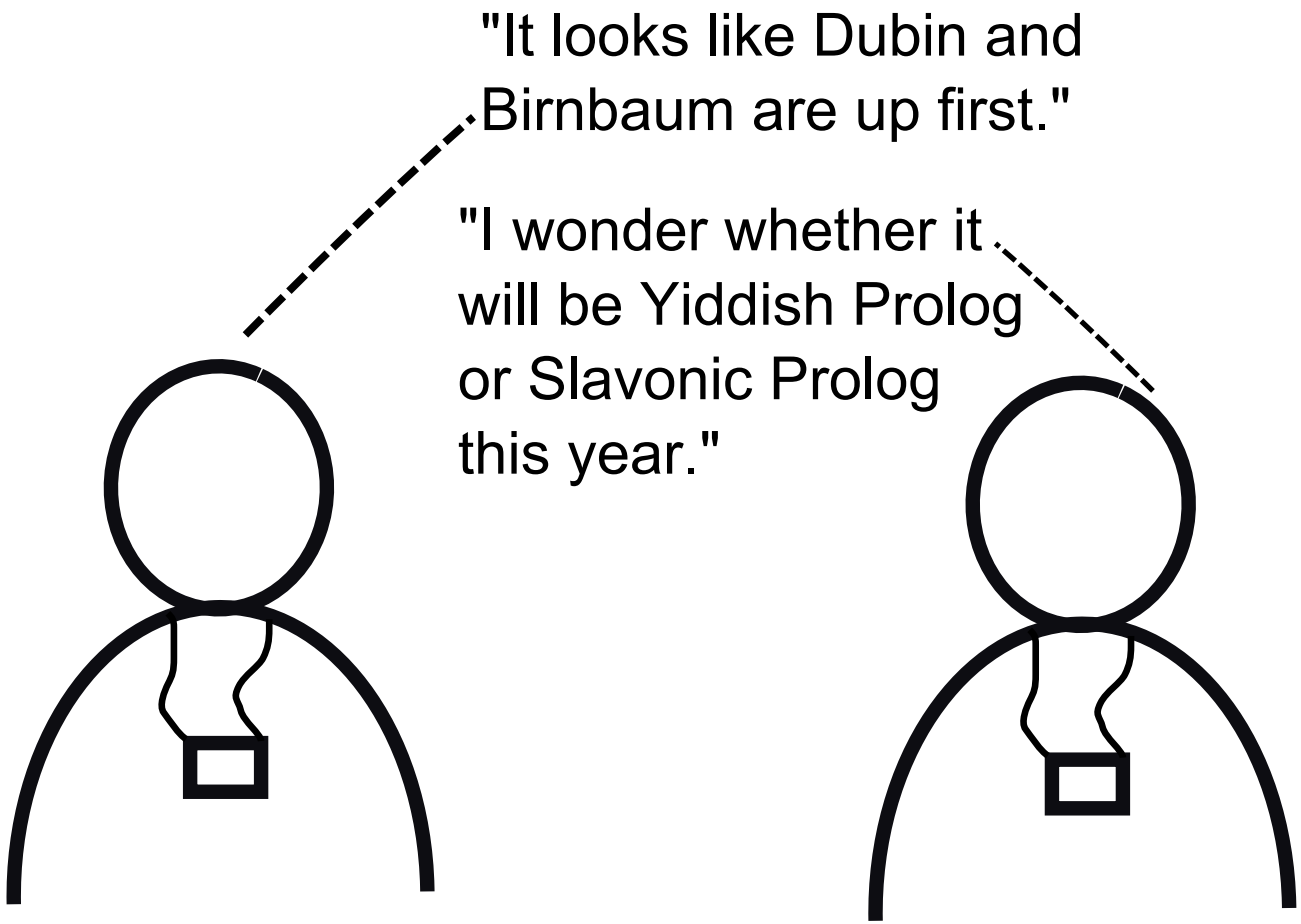
Reconsidering Conventional Markup as Knowledge Representation

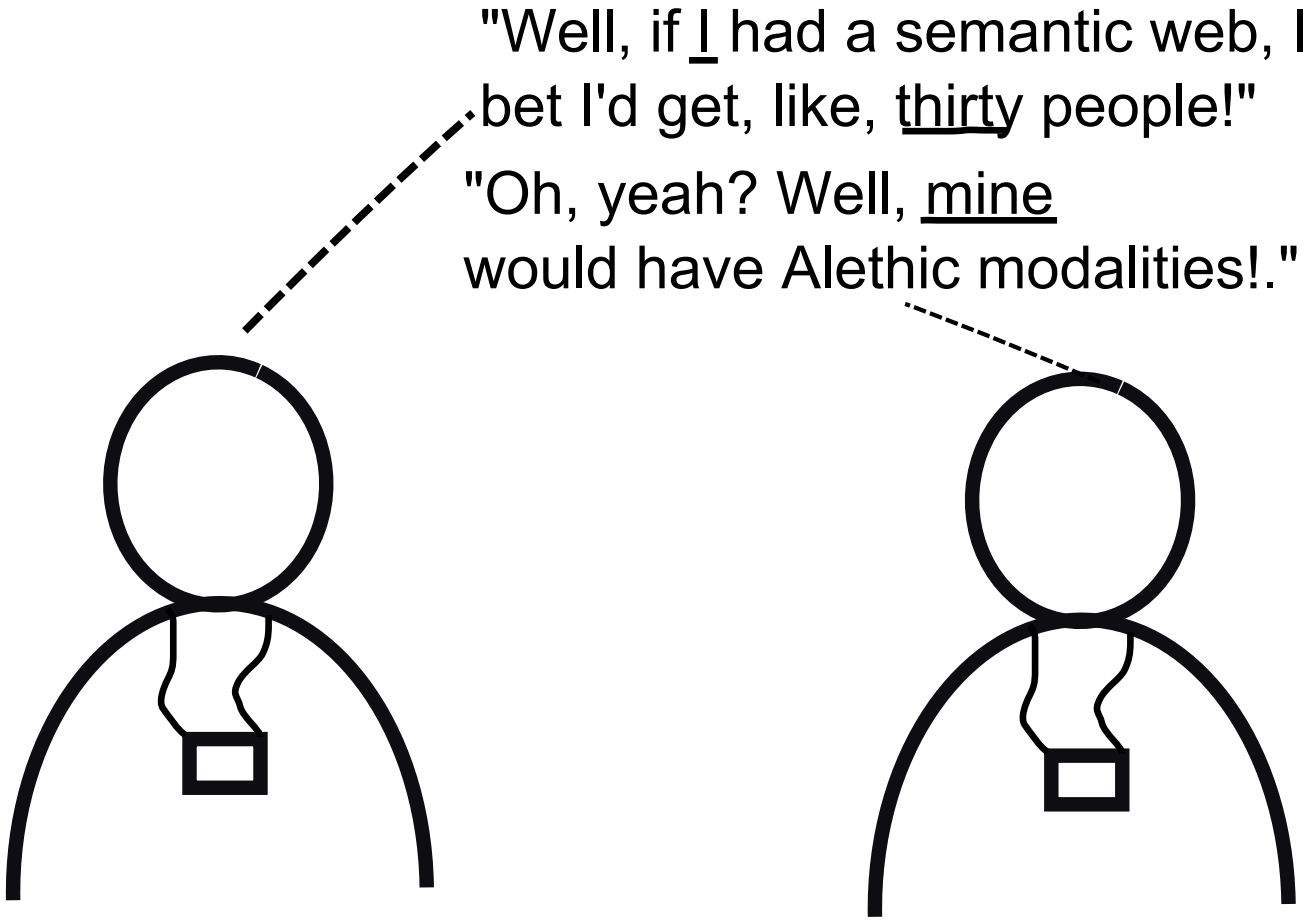
David Dubin and David J. Birnbaum

August 14, 2008

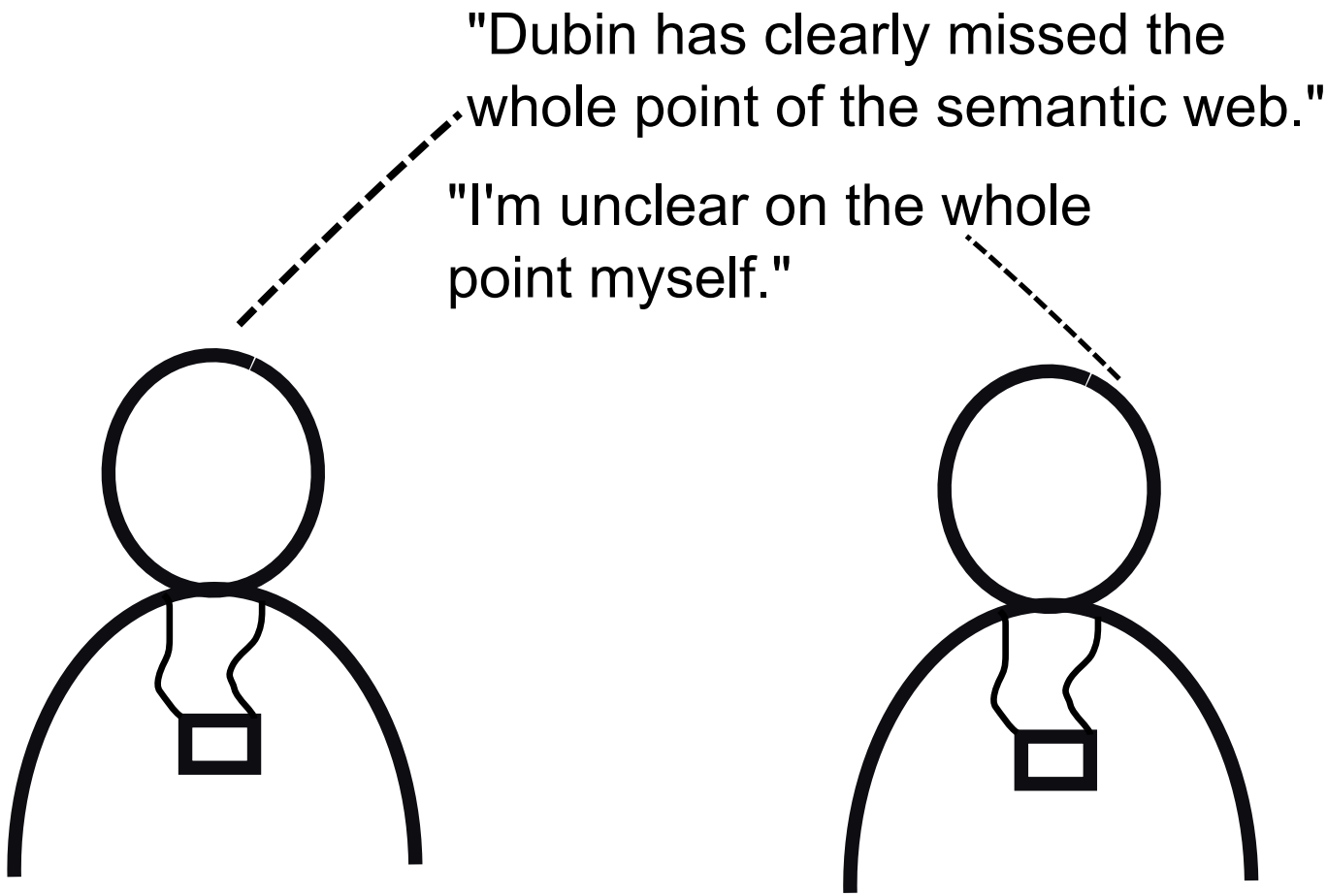
Our worry: before the talk



Preferable response: after the talk



More likely response (also fine with us)



Deductive Applications

- Inference or deduction can be a more appropriate processing style than procedural programming in many applcations.
- Running example: encoding theories of textual transmission.
- We're working on some complex inferences, but let's focus today on a few simple ones.

> *Otherwise, if we take the youngest common ancestor of the nodes that share a reading, we need to examine whether all descendants of that ancestor share that reading, or whether some of its descendants have the other reading.*

Example: A Stemmatology Application

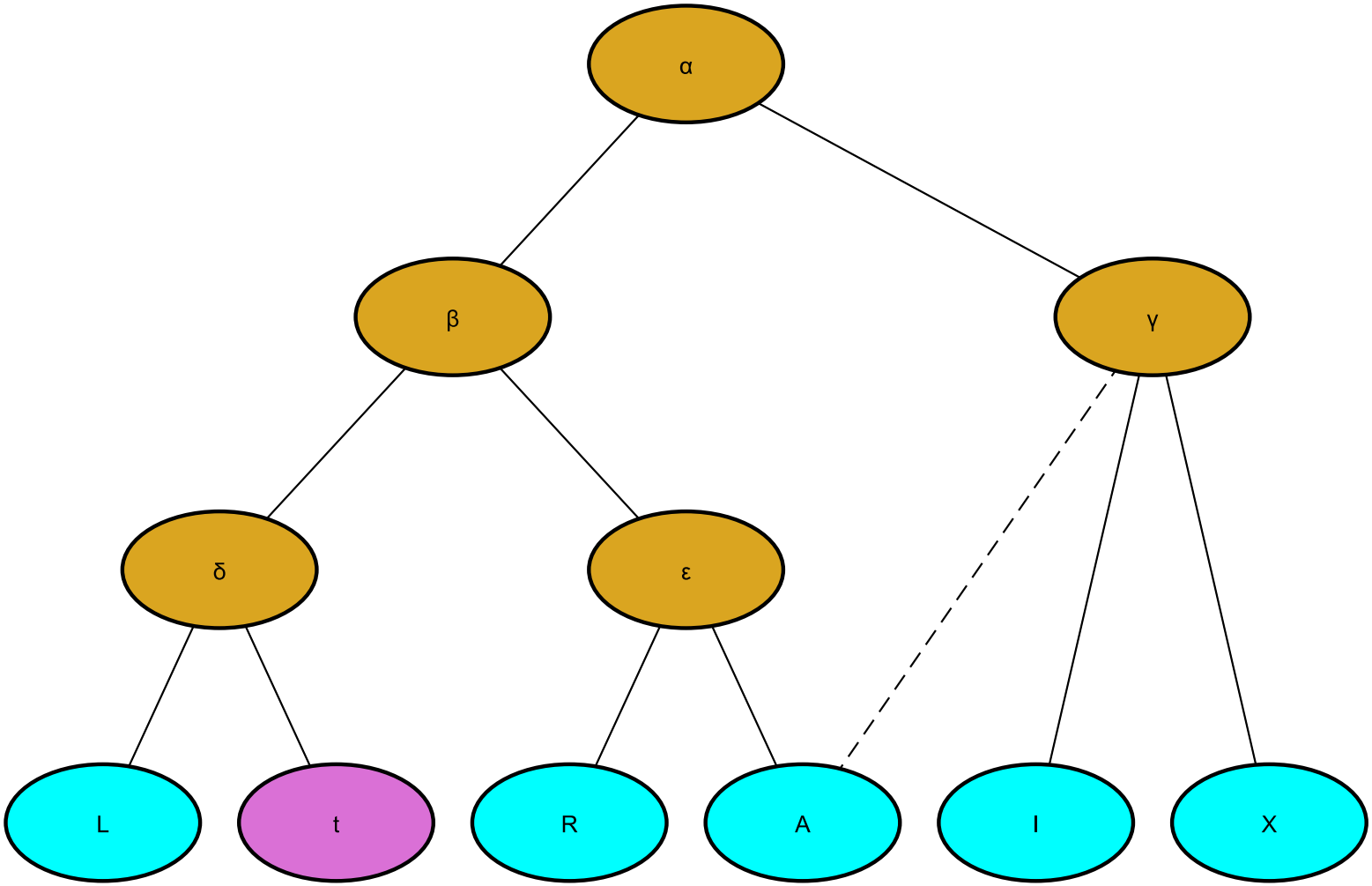


Figure: Stemma Codicum

Conventional XML Vocabulary

```
<node n="α" type="hypothetical">
    <node n="β" type="hypothetical">
        <node n="δ" type="hypothetical">
            <node n="L" type="extant"/>
            <node n="t" type="lost"/>
        </node>
        <node n="ε" type="hypothetical">
            <node n="R" type="extant"/>
            <node n="A" type="extant"/>
        </node>
    </node>
    <node n="γ" type="hypothetical">
        <contaminates target="A"/>
        <node n="I" type="extant"/>
        <node n="X" type="extant"/>
    </node>
</node>
```

Figure: Stemma Codicum

Some Attractions of RDF and OWL

- RDF, RDFS, and OWL have well-defined semantics.
- They offer various deductive frameworks (SWRL, DLP, etc.).
- Conventional markup is often semantically ambiguous.
- We can encode our stemmata either as conventional XML or in RDF.
- The expressive differences are seen more in the rule languages (e.g., Prolog vs. DLP).

RDF Expression

```
<eprg:ms eprg:status="hypothetical">
    <rdfs:label xml:lang="en-US">Alpha</rdfs:label>
    <eprg:derives>
       <eprg:ms eprg:status="hypothetical">
         <rdfs:label xml:lang="en-US">Beta</rdfs:label>
         <eprg:derives>
           <eprg:ms eprg:status="hypothetical">
            <rdfs:label xml:lang="en-US">Delta</rdfs:label>
            <eprg:derives>
                <eprg:ms eprg:status="extant">
                  <rdfs:label xml:lang="en-US">L</rdfs:label>
                </eprg:ms>
            </eprg:derives>
            <eprg:derives>
              <eprg:ms eprg:status="lost">
                <rdfs:label xml:lang="en-US">t</rdfs:label>
              </eprg:ms>
            </eprg:derives>
           </eprg:ms>
         </eprg:derives>
         <eprg:derives>
           <eprg:ms eprg:status="hypothetical">
            <rdfs:label xml:lang="en-US">Epsilon</rdfs:label>
            <eprg:derives>
              <eprg:ms eprg:status="extant">
                <rdfs:label xml:lang="en-US">R</rdfs:label>
              </eprg:ms>
            </eprg:derives>
```

Figure: Stemma Codicum in RDF

Expressive power vs. computational tractability

- We've known for years that small expressive differences in a KR formalism can make a big difference in whether an inference (e.g., class subsumption) is tractable or not.
- Full first order logic is only semidecidable.
- A great deal of worthy effort is devoted to finding decidable subsets of FOL, and developing scalable inference procedures.
- So if you can work within their constraints...

Description Logic Programming)

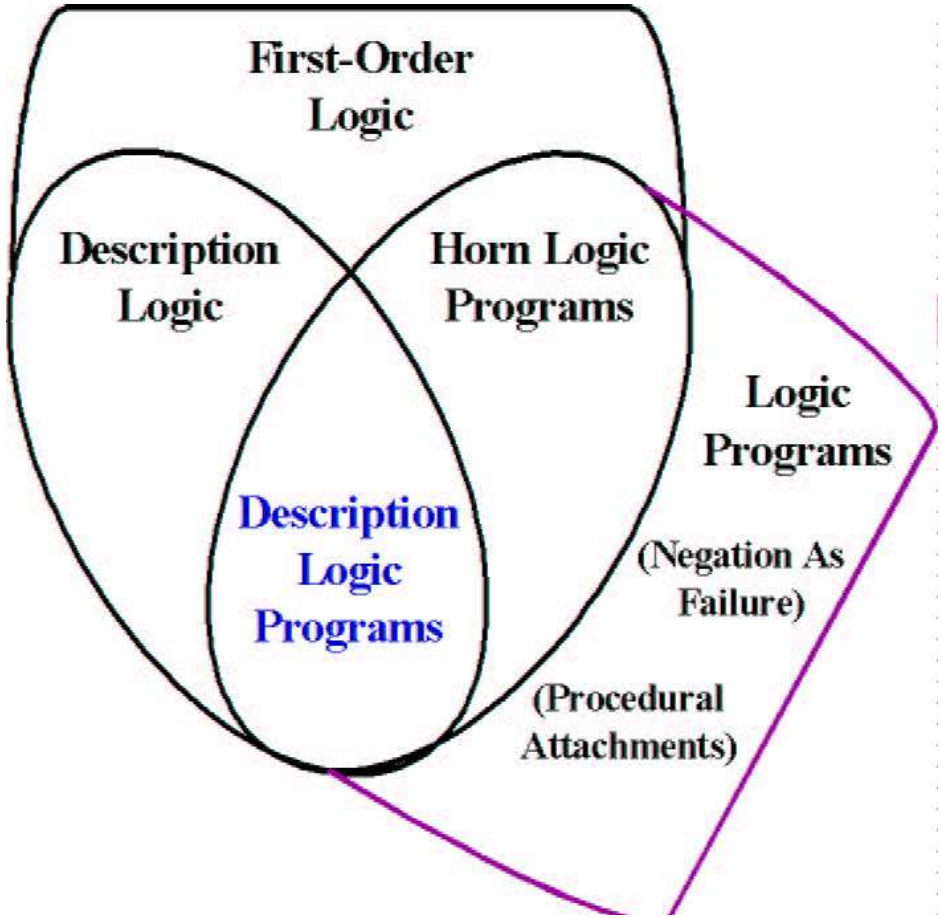


Figure: From Grosof, Horrocks, Volz, and Decker (2003)

Back to our example. This shouldn't be hard (and isn't).

```
/* Find Z, the youngest common ancestor of X and Y */
common_anc(X,Y,Z) :- parent(X,Z), parent(Y,Z).
common_anc(X,Y,Y) :- ancestor(X,Y).
common_anc(X,Y,X) :- ancestor(Y,X).
common_anc(X,Y,Z) :- not(siblings(X,Y)),
                     not(ancestor(X,Y)),
                     not(ancestor(Y,X)),
                     parent(X,W), parent(Y,V),
                     common_anc(W,V,Z).
```

Now find the youngest common ancestor for all manuscripts attesting a variant:
$\forall v\ \mathit{variant}(v) \supset \exists o\ \mathit{attests}(o, v) \cap [\forall x\ \mathit{attests}(x, v) \supset \mathit{ancestor\_or\_self}(x, o)]$

```
common_anc([X,Y], Z) :- common_anc(X,Y,Z).
common_anc([X|Tail], Z) :- common_anc(Tail, Y),
                           common_anc(X,Y,Z).
```

The Semantic Web

The Semantic Web seems to us to be about:

1. Knowledge-based, deductive applications
2. that act on structured data expressed in markup,
3. that are shared over the World Wide Web,
4. that draw on and reconcile numerous different ontologies,
5. and employ inference procedures that are guaranteed to scale (and to halt).

What if we stepped back from just the last two of those?

Summary

- Our applications require more expressive power than (for example) SWRL or DLP are offering.
- Why do theoretically unreliable inference procedures work well under many circumstances?
- If it's only because n is small in those circumstances, then that's not very interesting or encouraging.
- But sometimes it could be the structure of the data, rather than the size.
- Consider the linear strategy for forward chaining proof by resolution.

Conventional Markup

- Our RDF expression converts to logical assertions a little more directly than the XML expression.
- But defining formal semantics and inference procedures for conventional XML vocabularies is not difficult if you're willing to invest as much effort as an RDFS or OWL schema designer.
- Michael Sperberg-McQueen compares two methods in our 2002 paper.
- Or transform from application-specific XML to RDF, if you want.
- The term "microworld" has had pejorative connotations in the past.
- But developers of XML processing applications rely on focused, semantically well-behaved microworlds every day.