OBOL: Open Bio-Ontology Language

Using grammars to extract and use implicit knowledge in the GO and OBO

Chris Mungall

GO Advisors Meeting
Denver, CO
May 2004

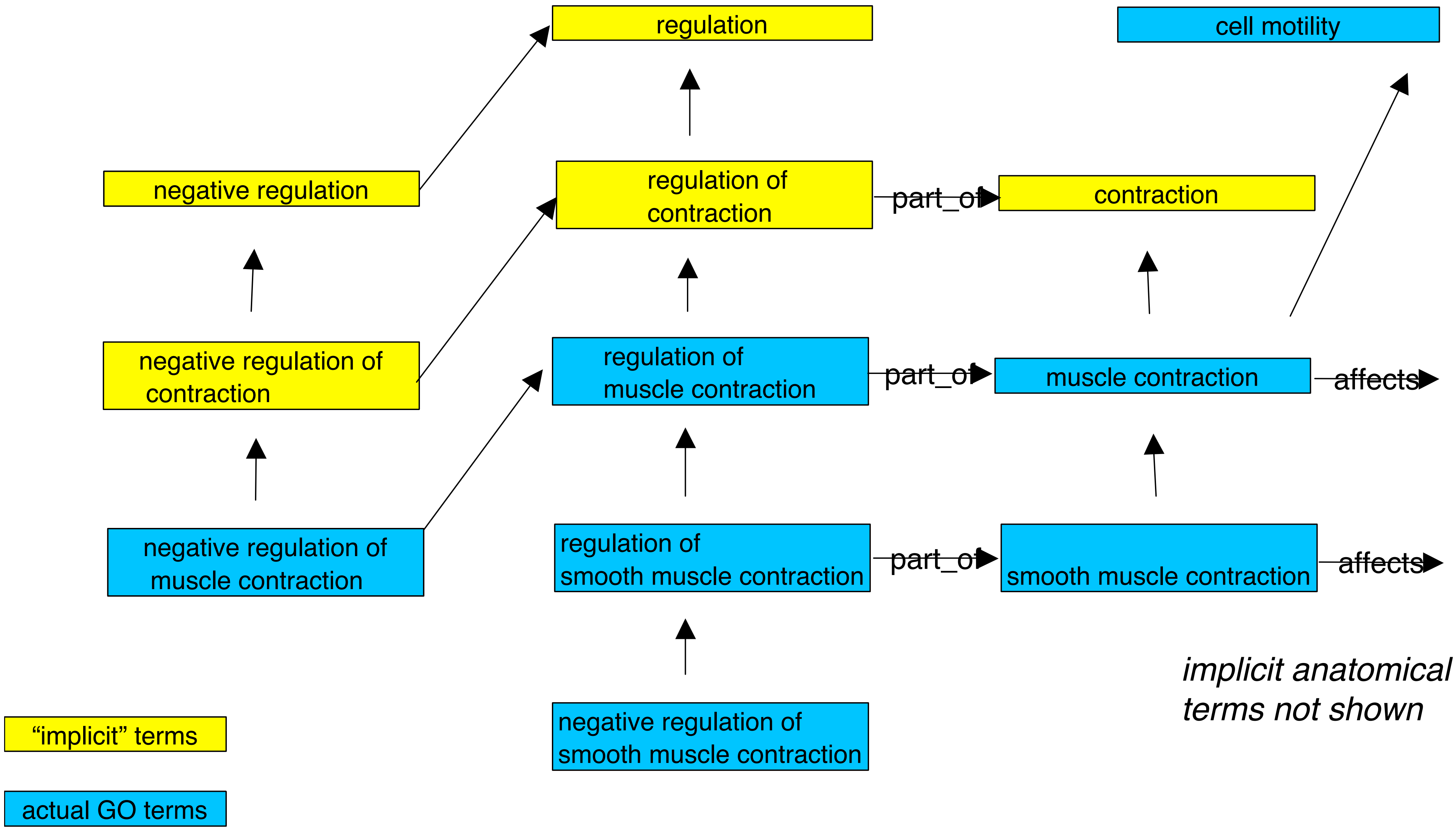
Outline

- Motivation – combinatorial issues with composite terms
- Approaches: annotation-time term composition *vs* tools for maintenance of large DAGs
- The OBOL System
  - Term decomposition using grammars
  - Generating computable logical class definitions
  - Rules and reasoning over class definitions
- Initial Results
- Strategies for using OBOL within GO/OBO

Manual maintenance of GO

- GO is 3 DAGs of over 16k terms
- Large DAGs of terms are hard to maintain
- "cross-products" produce combinatorial explosions and highly connected sub-graphs
- GO terms include OBO terms
  - eg `oxygen binding`; `wing development`
- Zipf's Law
  - Many terms not yet used in annotation (Ogren, pers. Comm.)

Example combinatorial explosion



One Approach: Properties

- One <u>extreme</u> solution is to **remove composite terms from ontology altogether**
  - Generate “anonymous” composite terms at annotation-time via *property/slot restrictions to `atomic terms`*
    - `binding` ^ *affects(*`interleukin-18`)
    - `contraction` ^ *affects(*`muscle`+*type*(`smooth`))
  - These bindings constitute a *class definition*
  - But it is still necessary to make statements about composite terms in the ontology
    - `macrophage activation` *is_a* `immune cell activity`
    - `fibrinolysis` *is_a* `negative regulation of blood coagulation`

Another approach: *Computationally aided ontology maintenance*

- GO terms exhibit regularity in their *syntactic structure*
  - *Substring* relationships highly correlated with actual relationships
    - regulation of smooth muscle contraction
    - smooth muscle contraction
    - muscle contraction
    - contraction

Ogren PV, Cohen KB, Acquaah-Mensah GK, Eberlein J, Hunter L. 2004. **The compositional structure of Gene Ontology terms.** *Pac Symp Biocomput 9: 214-215.*

OBOL: Syntax and Semantics

- What about using the term syntax to get at the *meaning* of the term?

GO/OBO Term

↕

Class Definition
may involve relationships to other OBO terms

↓

Inferences

“`interleukin binding`”

↕

`binding`^*affects*(`interleukin`)

↓

“`interleukin binding`” *is_a* “`cytokine binding`”
inferred from: interleukin *is_a* cytokine

How OBOL Works

- Term names are parsed using a *grammar*, generating parse trees
- Parse trees are turned into class definitions using transformation rules and *property definitions*
  - Both steps are reversible
- Inferences are made on the class definitions
- Implemented in Prolog

Computational Grammars

- A collection of **transformation rules** for parsing (**decomposing**) and generating (**composing**) *sequences of symbols (ie words).*
- *A language grammar operates on words, which must be categorised into word senses.*
  - *e.g. noun, adjective, preposition*
- *GO/OBO term grammars require very few word senses*

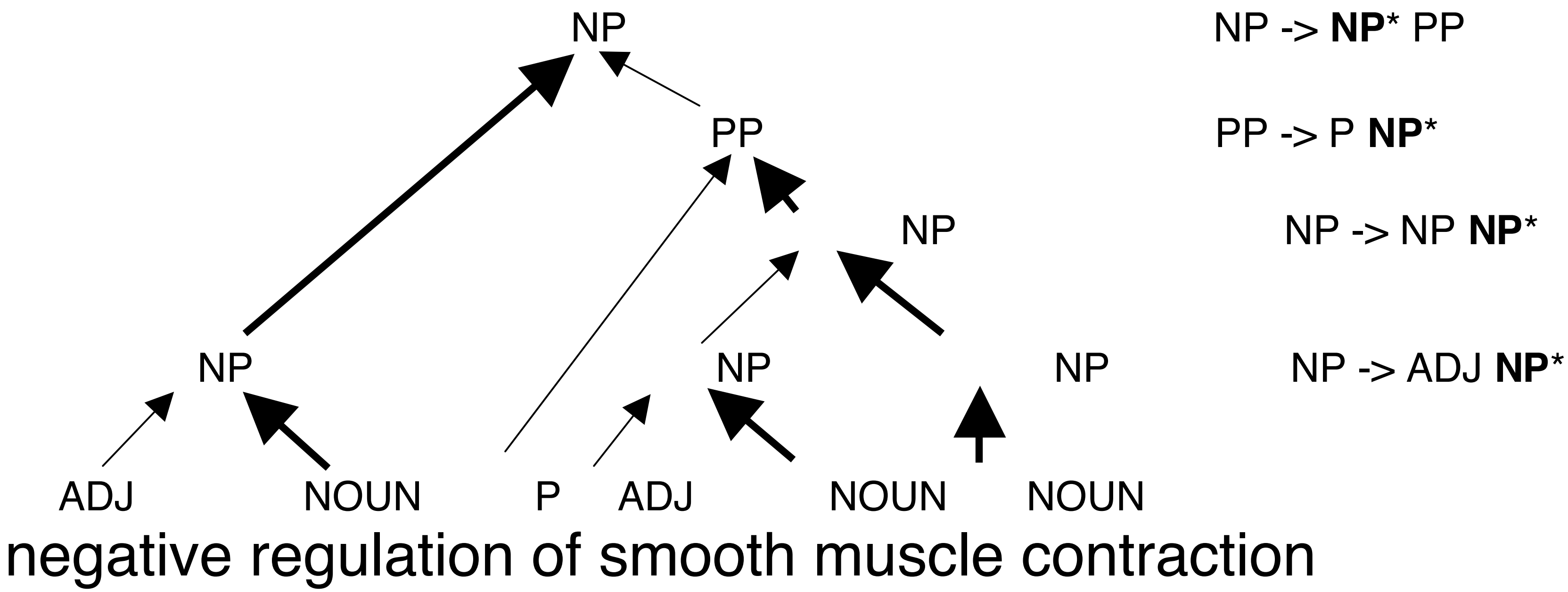
A simple OBO term grammar

This is a subset of the whole OBO grammar:

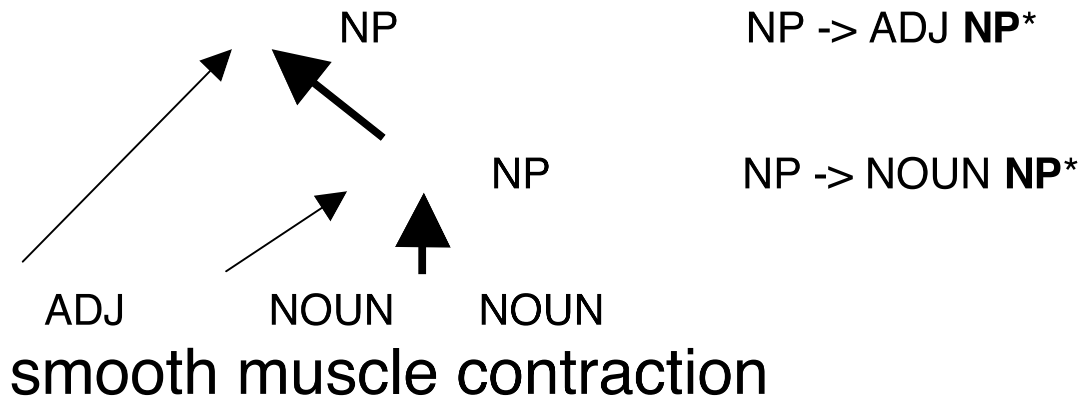
Term	--> NP	*e.g.* negative regulation of smooth muscle conraction
NP	--> ***NP*** PP	*e.g.* ***negative regulation*** of smooth muscle conraction
NP	--> NOUN	*e.g.* muscle
NP	--> ADJ ***NP***	*e.g.* smooth ***muscle***
NP	--> NP ***NP***	*e.g.* smooth muscle ***contraction***
PP	--> PREP ***NP***	*e.g.* of ***smooth muscle contraction***

Concrete nouns are treated the same way as abstract nouns (*contraction* is treated as a noun, even though it is the inflecte form of the verb *contract*)

A Parse Tree for a GO Term



*alternate parses are possible (parse **forests**)*



Atomic ontologies: OBO *wordlists*

- A term grammar requires a **vocabulary** of words
- These words correspond to atomic terms from the OBO ontologies
- Currently the wordlists are generated semi-automatically
- Relational adjectives are paired with the appropriate noun
  - [cytosol*ic*, cytosol], [coat*ed*, coat]
- OBOL exhibits graceful degradation with incomplete wordlists
  - unrecognised words treated as *orphan nouns*

Making *Logical Class Definitions* from Parse Trees

- A Class definition is a compound term with *property/slot* restrictions
  - `contraction`^*affects(*`muscle`^*type*(`smooth`))
- *A class definition can be exported using either OBO or OWL format*
- *A class definition can be generated from a parse tree using transformation rules and property/slot* ***definitions***

Properties guide class tree building

Property: **affects_cell_type**
domain: *biological_process*
range: *cell_type*
context: *modifier(np)*

Muscle contraction

Contraction^*affects*(muscle)

Property: **regulates**
domain: *regulation*
range: *biological_process*
context: *preposition(of)*

Regulation *of* muscle contraction

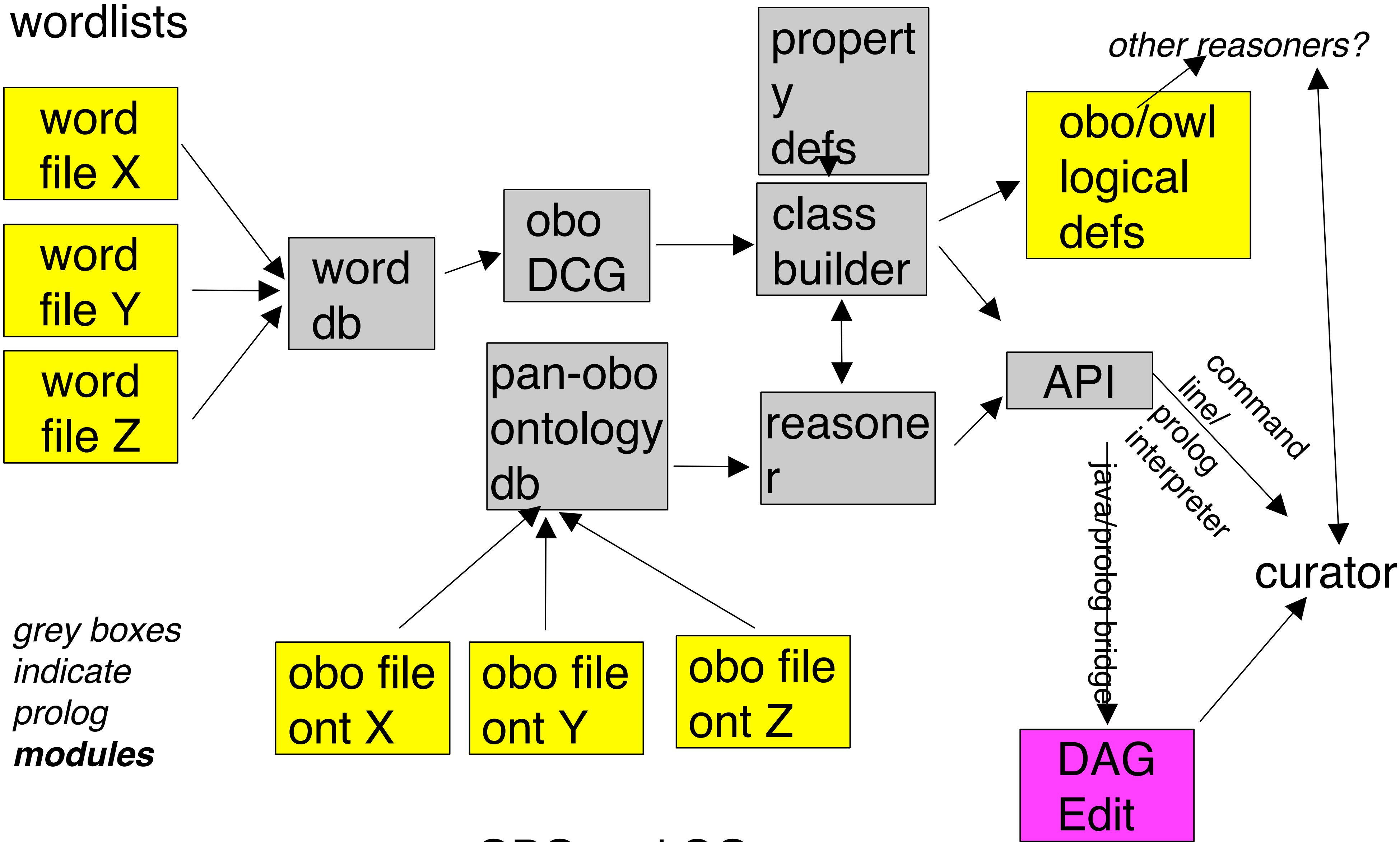
Regulation^*regulates*(contraction^*affects*(muscle))

The property definitions above constrain how one class (the domain, or *su*
can **relate** to another (the range, or *object*) in a given grammatical context

Reasoning over class definitions

- We can use classdef rules to...
  - place new terms in the correct place in the DAG
  - check for missing relationships in the DAG
  - find inconsistencies between ontologies
- Method:
  - Inference rules implemented in Prolog
  - Interactive use or as DAG-Edit plugin
  - *OR* export to OWL and use Protege + Racer [not tested yet]

OBOL Architecture



OBO and GO

Initial Results

From biological_process and cellular_component only

(3630/9067 have unique parses)

derivable	3247	*substring*	1828
		NOT-a-substring	1419
non-derivable	10445	*substring*	2285
		NOT-a-substring	8160
suspected-missing	400	*substring*	379
		NOT-a-substring	21

Example suspected missing relationships:

nucleolar chromatin *part_of* nucleolus
clathrin-coated vesicle *has_part* clathrin coat
chromoplast membrane *is_a* plastid membrane
nuclear microtubule *part_of* nucleus
vitamin E biosynthesis *is_a* vitamin E metabolism

OBOL doesn't currently check for inverses!

Using OBOL with GO/OBO

- The complete OBOL system can be implemented within GO/OBO in a variety of ways
  - "Behind the scenes"
    - GO curators maintain same mode of working and receive periodic auto-generated reports
  - Within DAG-Edit
    - GO curator uses OBOL interactively
  - To transition GO to a more "formal" ontology

(1)Using OBOL Behind the Scenes

- Maintain facade of *narrative* approach, whilst implementing a *combinatorial* approach behind the scenes
  - Curators carry on working their current mode
  - OBOL is used periodically to check the ontology
  - suggested edits are submitted to curators en-masse
  - OBOL is occasionally invoked on-demand to produce a new subgraph of cross-products (eg development vs anatomy)
- Longer feedback cycle is less efficient
- We are ready to go in this mode *now*

(2)Using OBOL from DAG-Edit

- OBOL is invoked by curators from DAG-Edit
  - suggested corrections can be highlighted
  - new composite terms can be automatically placed in the DAG
  - Errors can be spotted immediately
- *Slot/property* based annotation
  - non-curators producing annotations (instances) can create new classdefs on-the-fly
  - OBOL can check their validity and automatically create the subsumption path

(3)Using OBOL to recast ontologies

- OBOL is used as a one-off to help generate classdefs for all terms in an ontology
  - classdefs are then maintained by curator
  - Subsequent uses of OBOL are in reverse-mode; automatic generation of term names from classdefs
  - OBOL or other reasoner invoked from DAG-Edit to spot mistakes and generate subsumption paths
  - DAG-Edit and OBO format now supports classdefs (aka *complete* definitions)
  - Major transition; more or less work for curators?

Problems to address

- Parsing issues: chemicals, wordy terms
- Logic issues: sensu
- Supporting OBO orthogonal ontologies
    - Difficulties with biochemical ontology; plurals
    - No generic anatomy ontology (as yet)
    - No protein/complex ontology
  - Can we use OBOL to help construct these other ontologies?
    - *YES*

What next?

- Better coverage:
  - refine slot/property definitions
- Grammar for human-readable text definitions
- Extend inference rules??
  - e.g. Non-monotonic reasoning (cell HAS-PART nucleus EXCEPT erythroctye)
- Generate OWL from derived class definitions
  - distribute (with caveats) to logic community
  - Use protege+racer as reasoner

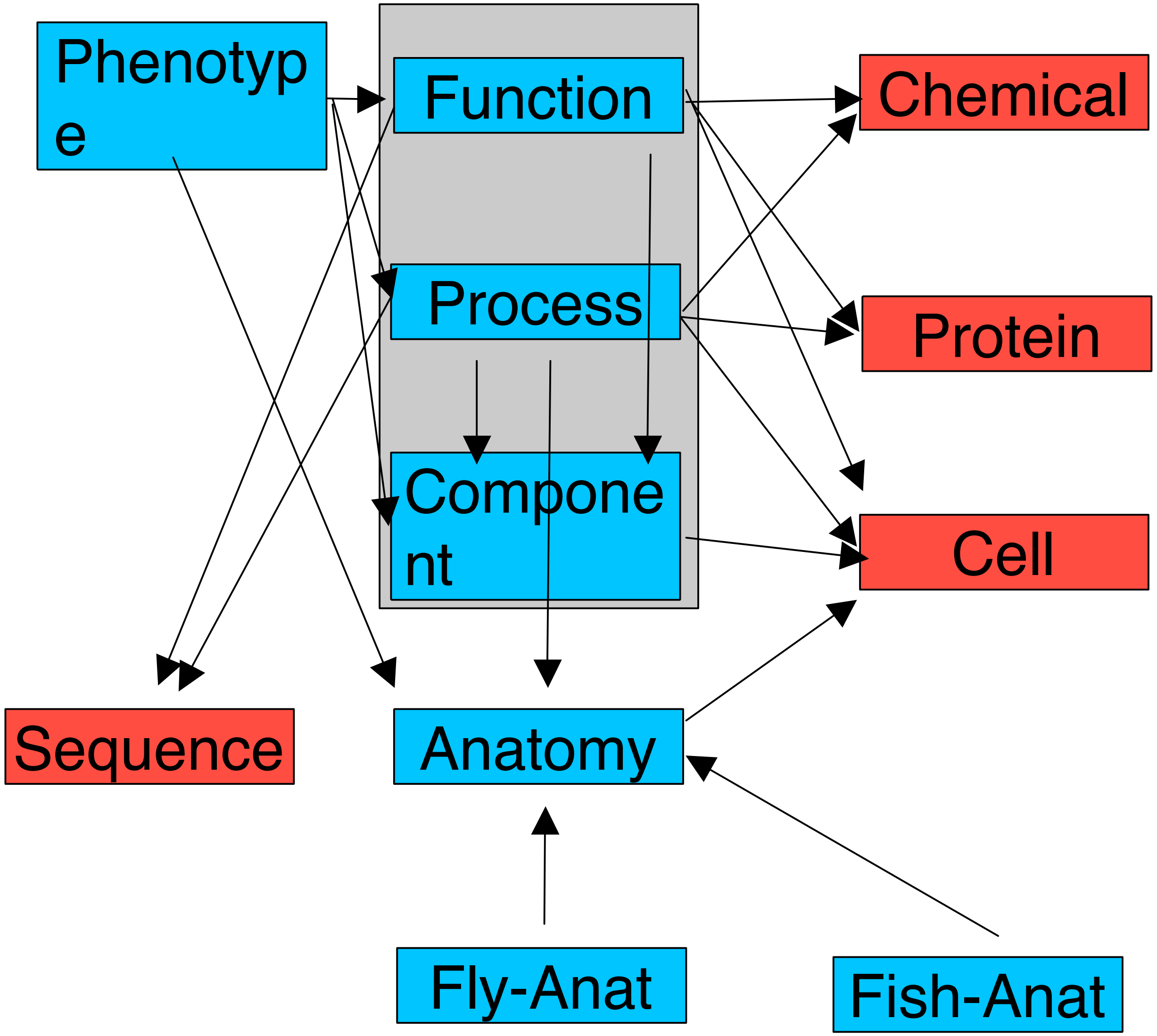
Conclusions

- OBOL can help with the combinatorial explosion in a number of ways
- Initial results with incomplete wordlists and property definitions are promising
- Combining a term grammar with reasoning is powerful and offers significant advantages over either purely syntactic or semantic approaches
- Should OBO focus more the *atomic* units of the ontologies?

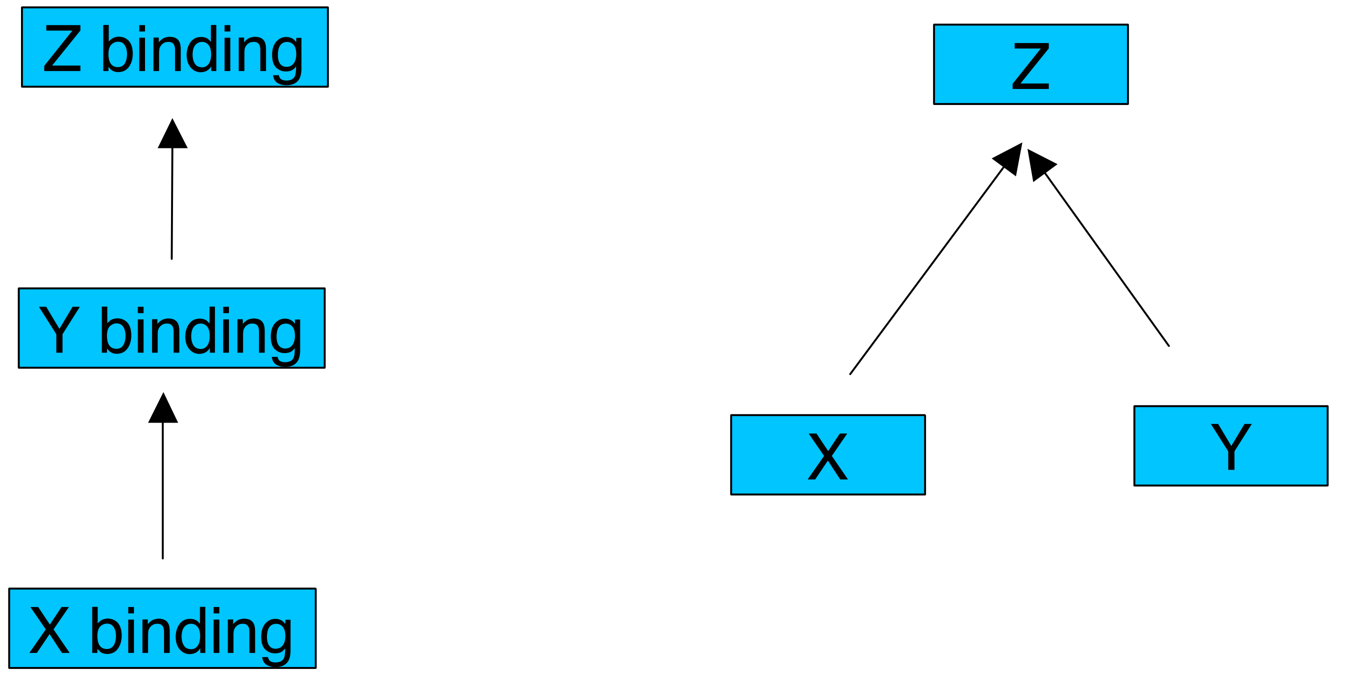
Acknowledgements

- Suzanna Lewis
- John Richter
- Brad Marshall
- Karen Eilbeck
- David Hill
- Joel Richardson
- GO Curators
- Michael Ashburner
- Chris Wroe
- Robert Stevens

The OBO Universe (partial)



Detecting inconsistencies



Prolog as an ontology language

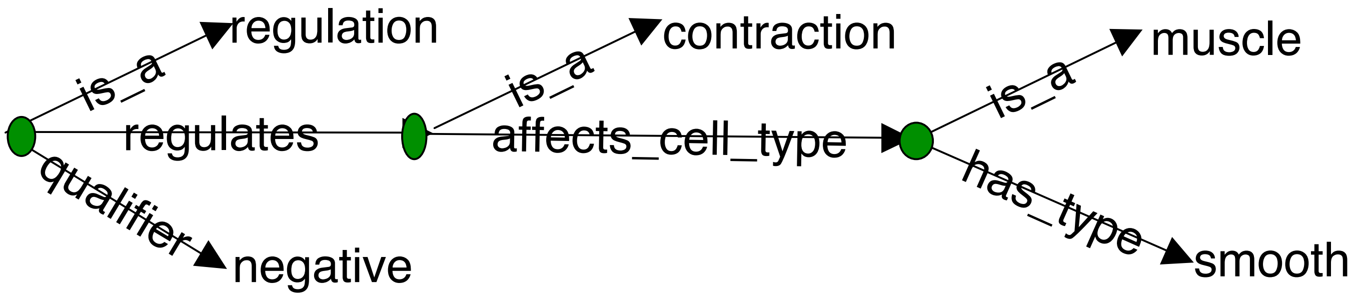
- **% DATABASE OF FACTS**
- isa(carb_binding, binding).
- isa(polysac_binding, carb_binding).
- isa(chitin_binding, polysac_binding)
- isa(cellulose_binding, polysac_binding).

- **% INFERENCE RULES**
- isaT(X,Y):- isa(X,Y).
- isaT(X,Y):-isa(X,Z), isaT(Z,Y).

- ?- isaT(chitin_binding, binding).
- **YES**
- ?-isaT(X, polysac_binding).
- **X=carb_binding.**
- **X=chitin_binding.**
- **X=cellulose_binding.**
- ?-isaT(chitin_binding, cellulose_binding).
- **NO**
- **?-isaT(X,Y).** ***% returns all paths***

Prolog internal representation

class(regulation <process>
qualifier=class(negative <general>)
regulates=class(contraction <process>
affects_cell_type=class(muscle <anatomical>
has_type=class(smooth
<general>))))



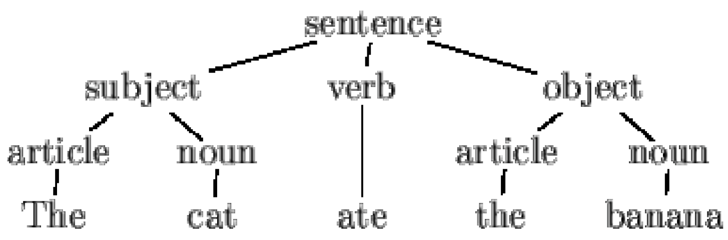
Prolog Grammar Implementation

- Prolog: the classic logic programming language
  - High-level declarative language, natural choice for ontologies; built in “database”
  - Definite Clause Grammars (**DCG**s)part of the language; DCGs allow passing data up the parse tree
- XSB Prolog
  - Uses **tabling** (more efficient, less re-calculation)
  - Tabling + DCGs = *chart parsing* (Earley's algorithm)

A Formal Grammar for OBO terms

- All(?) GO/OBO terms are NOUN-PHRASES (exception: phenotypes?)
- A NOUN-PHRASE is (recursively) made from
  - a NOUN (includes inflected verbs; eg *bind<u>ing</u>*)
  - an ADJECTIVE followed by a NOUN-PHRASE eg *inner membrane*
  - a NOUN-PHRASE preceeded by a NOUN-PHRASE *acting as* ADJECTIVE; eg *clathrin coat*
  - a NOUN-PHRASE then PREPOSITION then NOUN-PHRASE; eg *regulation of transcription*
  - an (optional) NOUN-PHRASE then a RELATIONAL ADJECTIVE then a NOUN-PHRASE; eg *clathrin-coat<u>ed</u> vesicle*
- *Precedence rules* are also required to prune parse forest
- Simple but effective

Parse tree for a simple sentence,
"*the cat ate the banana*"

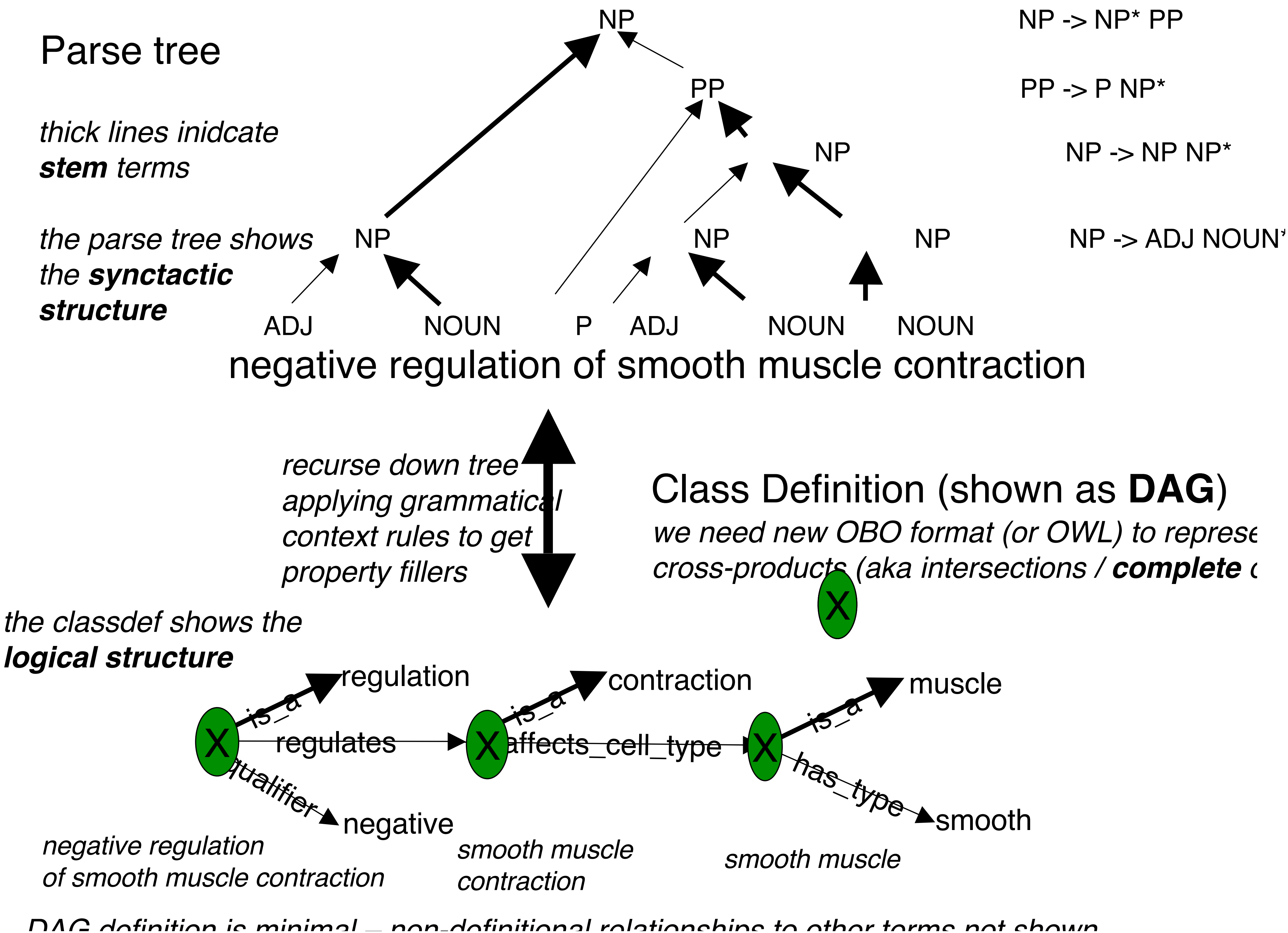


GENERATING: Start at top ('sentence')
and apply rules until all symbols are terminal

PARSING: Start at bottom – a sequence of terminals;
apply rules, combining symbols if necessary

Sentence -> Subject Verb Object
Subject -> Article Noun
Object -> Article Noun
Article -> a l the
Verb -> ate l chased
Noun -> cat l banana l mouse

- A formal grammar is a set of *production rules* operating over *terminal symbols* (eg words) and *non-terminal symbols (eg word/phrase categories)*
- The rules determine how sequences of symbols can be *transformed, making a parse tree*


Parse tree
thick lines inidcate
stem terms
the parse tree shows
the synctactic
structure
NP
PP
NP
NP
NP
NP
NP -> NP* PP
PP -> P NP*
NP -> NP NP*
NP -> ADJ NOUN*
ADJ
NOUN
P
ADJ
NOUN
NOUN
negative regulation of smooth muscle contraction
recurse down tree
applying grammatical
context rules to get
property fillers
Class Definition (shown as DAG)
we need new OBO format (or OWL) to represe
cross-products (aka intersections / complete d
the classdef shows the
logical structure
X
regulation
is_a
regulates
qualifier
negative
X
contraction
is_a
affects_cell_type
X
muscle
is_a
has_type
smooth
X
negative regulation
of smooth muscle contraction
smooth muscle
contraction
smooth muscle
DAG definition is minimal – non-definitional relationships to other terms not shown

COPII-coated vesicle membrane

class(membrane <component> “COPII-coated vesicle membrane”
part_of=class(vesicle <component> “COPII-coated vesicle”
has_part=class(coat <component> “COPII coat”
made_from=class(COPII <complex>
“COPII”))))

class/term name shown in quotes; these can be derived by reversion the transformation

the above classdef is consistent with what is in the GO `cellular_component` ontology

requires use of **inverse properties** (*has_part* vs *part_of*)
- supported in new OBO format.

Inference of intermediate terms and IS_As – example rule

FORALL classdef pairs **IFF** the stem-class is the same **AND** all the property-values in the restriction-list are identical **EXCEPT** for one property, in which the property-values are linked by an isa, **THEN** the classdefs are linked by an isa

class(regulation
process_regulated=R
qual=Q)
is_a
class(regulation
process_regulated=R'
qual=Q)
<=>
R is_a R'

class(C
P1=V1 P2=V2..Px=Vx
Pn=Vn)
is_a
class(C
P1=V1 P2=V2..Px=Vx'
Pn=Vn)
<=>
Vx is_a Vx'