

Ontologies Growing Up: Tools for Ontology Management



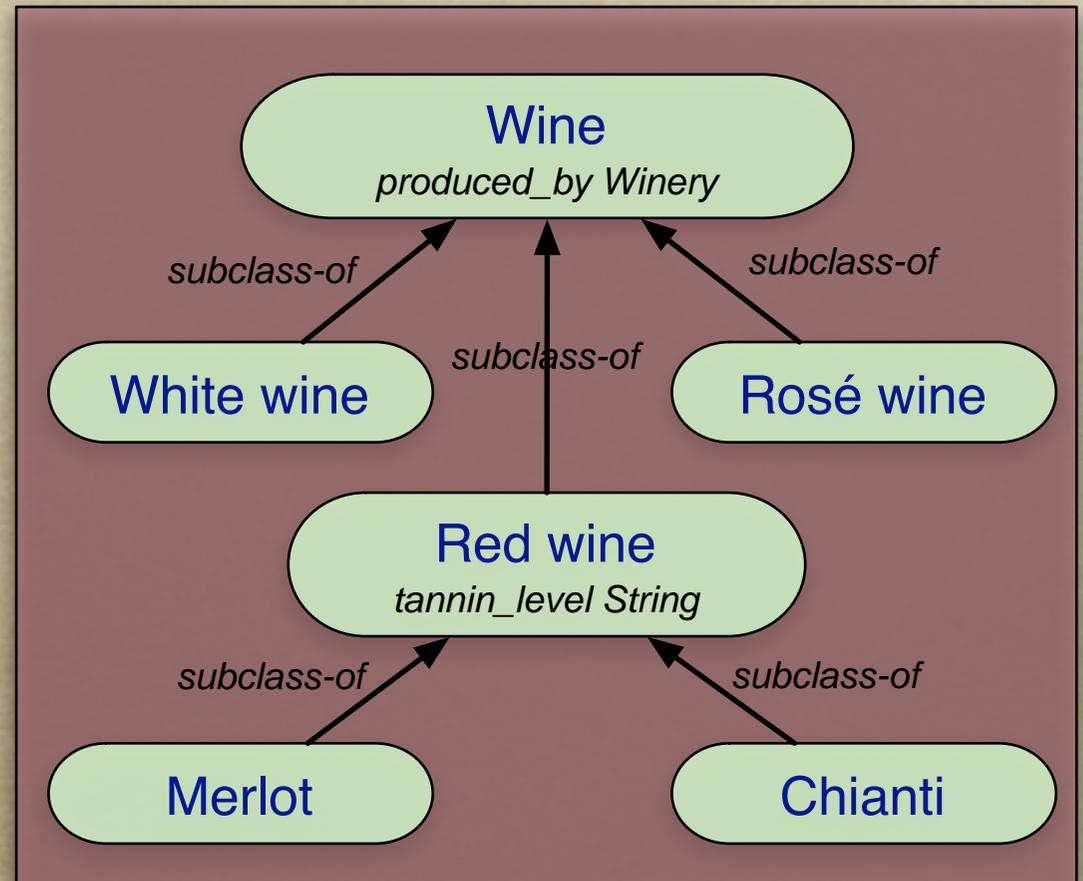
Natasha Noy
Stanford University

An ontology

- *Conceptualization of a domain that is*
 - *formal*
 - *can be used for inference*
 - *makes assumptions explicit*
 - *shared, agreed upon*
 - *enables knowledge reuse*
 - *facilitates interoperation among applications and software agents*

An ontology (II)

- Defines *classes*, *properties*, and *constraints* in a domain



The Good News

- *Ontologies are the backbone of the Semantic Web*
 - *More ontologies are available*
- *Ontology-development tools lower the barrier for ontology development*
 - *More people are developing ontologies*

The Good News I: Semantic Web

- *Ontology languages defined as standards: **RDF Schema** as **OWL***
- *A huge playing field for ontology research and practice*

More Good News: Ontology Tools



- *Ontology-development becomes more accessible*
- *Protégé*
 - *Developed at Stanford Medical Informatics*
 - *Is an **extensible** and **customizable** toolset for*
 - ***constructing** knowledge bases*
 - *developing **applications** that use these knowledge bases*



Classes Slots Forms Instances Queries

CLASS BROWSER

For Project: ● wines

Class Hierarchy

- ▶ ● Food
- ▼ ● Drink
 - ▼ ● Wine
 - ▼ ● White wine
 - ▶ ● White Burgundy
 - Chardonnay
 - Chenin Blanc
 - Pinot Blanc
 - Sauvignon Blanc
 - Semillon
 - ▶ ● Riesling
 - Sauterne
 - White Bordeaux
 - Sancerre
 - Muscadet
 - Ice Wine
 - ▶ ● Rose wine
 - ▶ ● Red wine

CLASS EDITOR

For Class: ● Wine (instance of :STANDARD-CLASS)

Name: Wine

Documentation: A wine class represents all possible wines

Constraints:

Role: Concrete ●

Template Slots

Name	Cardinality	Type	Other Facets
body	single	Symbol	allowed-values={FULL,MEDIUM,LIGHT}
color	single	Symbol	allowed-values={RED,ROS,WHITE}
flavor	single	Symbol	allowed-values={DELICATE,MODERATE,S
grape	multiple ...	Instance of Wine grape	
maker	single	Instance of Wine maker	
name	single		
sugar	single		

Chateau Morgon Beaujolais (instance of Beaujolais)

Name: Chateau Morgon Beaujolais

Body: LIGHT

Color: RED

Maker: Chateau Morgon

Flavor: DELICATE

Sugar: DRY

Grape: Gamay grape

Tannin Level: LOW

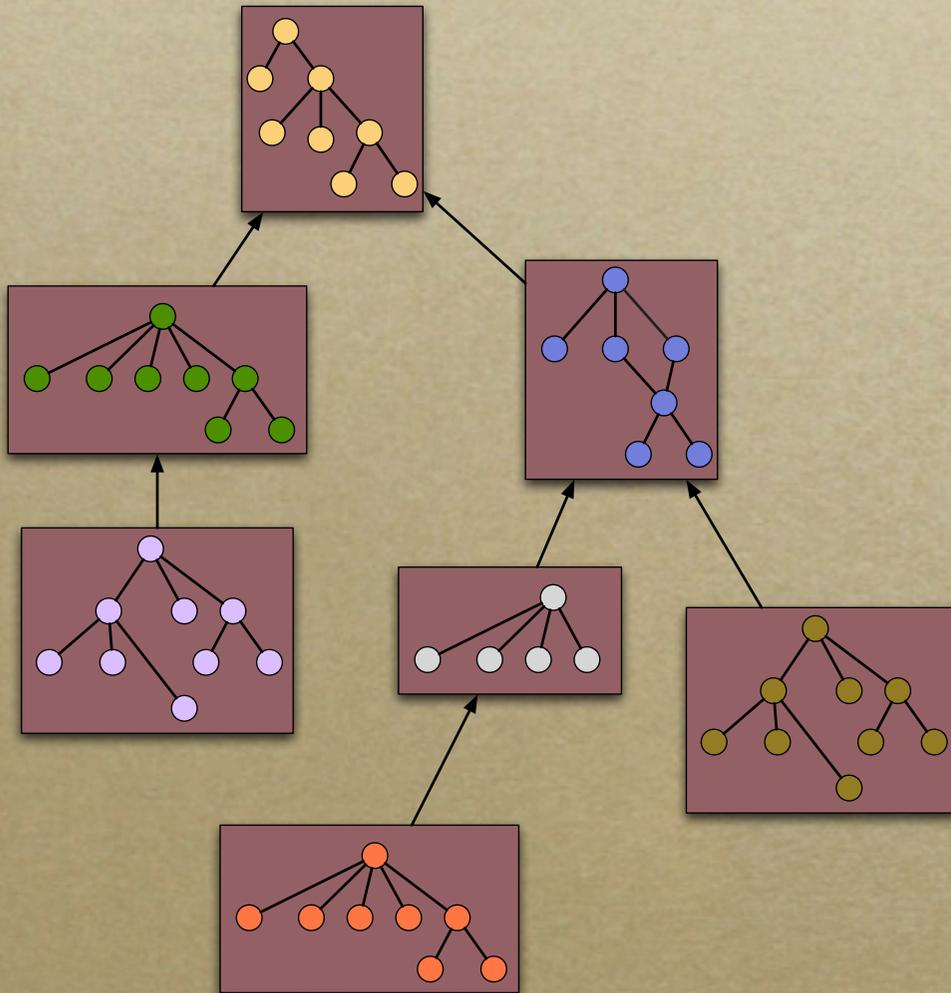
Protégé



- *What makes Protégé different?*
 - *Automatic generation of graphical-user interfaces, based on user-defined ontologies, for acquiring domain instances*
 - *Extensible knowledge model and architecture*
 - *Scalability to very large knowledge bases*
 - *Available under an open-source license*

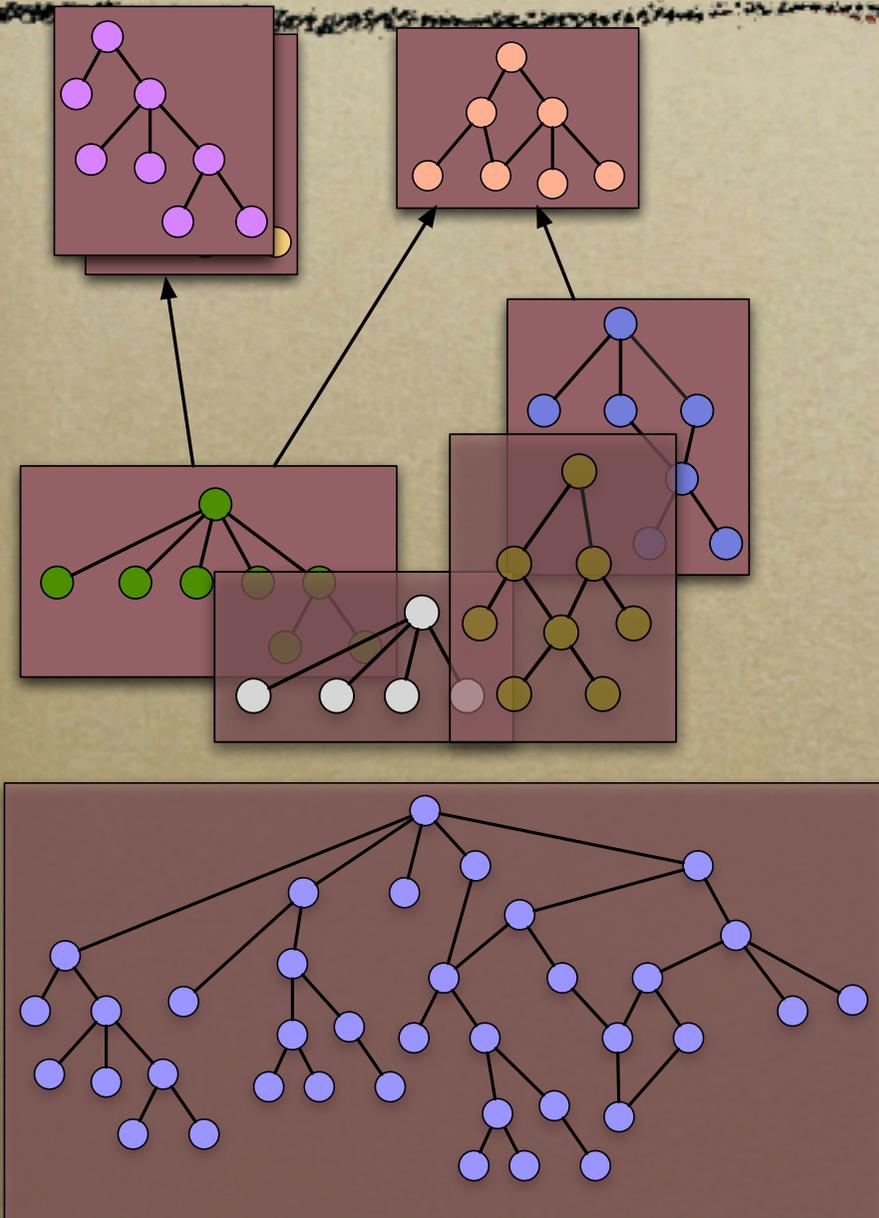
<http://protege.stanford.edu>

The Ideal World



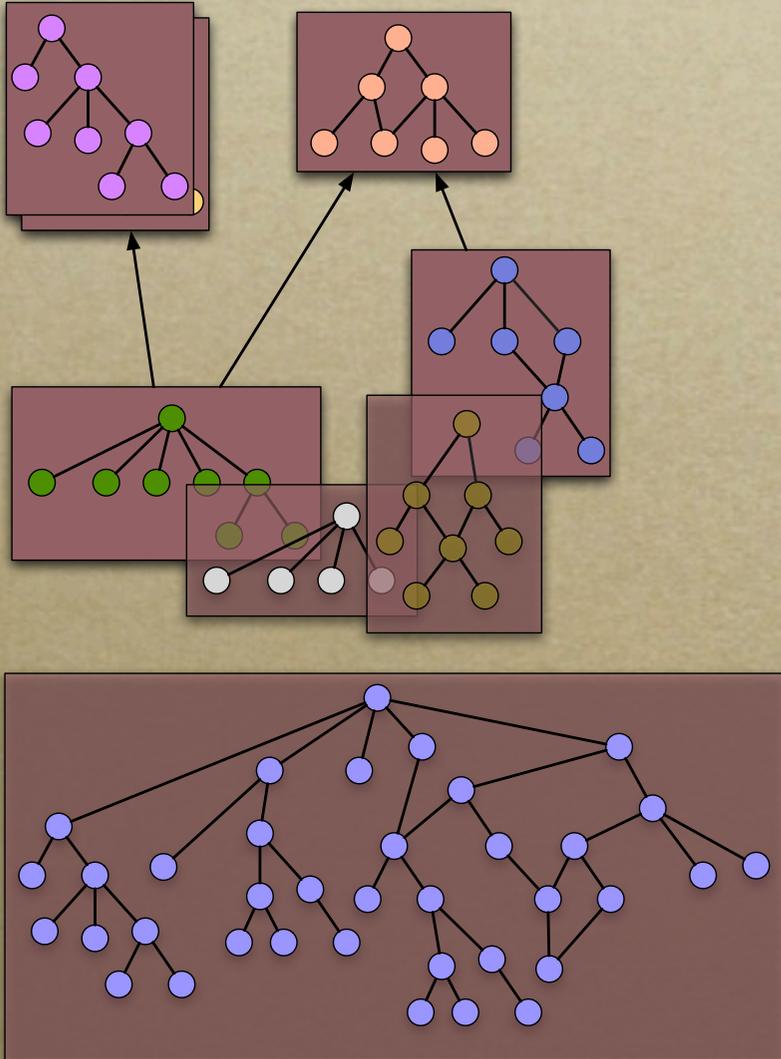
- The same *language*
- No overlap in *coverage*
- No new *versions*
- A single *extension tree*
- Small reusable *modules*

The “Bad” News: The Real World



- ~~The same language~~
- ~~No overlap in coverage~~
- ~~No new versions~~
- ~~A single extension tree~~
- ~~Small reusable modules~~

PROMPT: Dealing with the Messy World



- *Find similarities and differences between ontologies*

ontology mapping and merging

- *Compare versions of ontologies*

ontology evolution

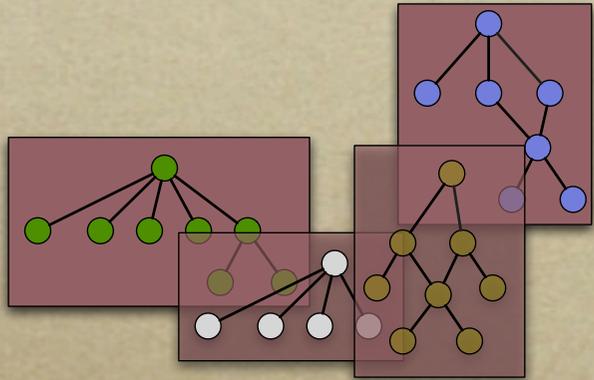
- *Extract meaningful portions of ontologies*

ontology views

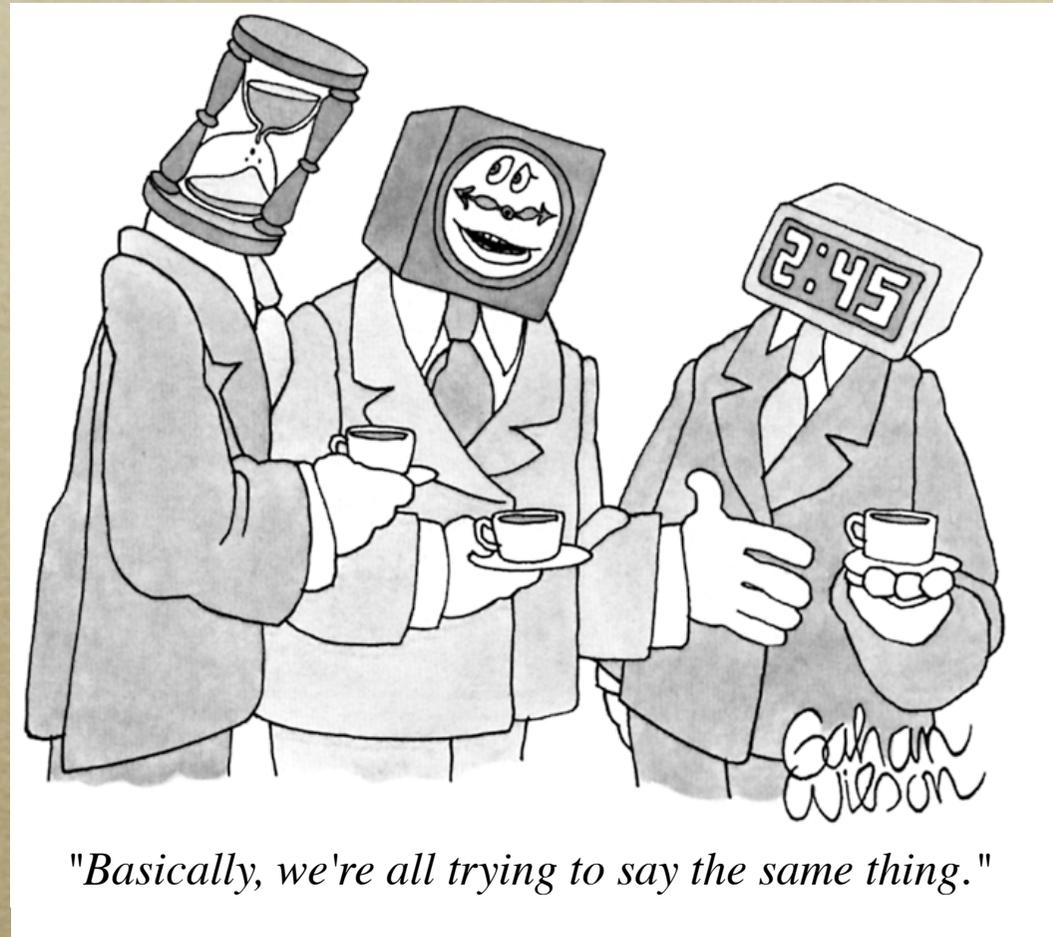
- *Integrate in an ontology-editing environment*

Protégé plugin

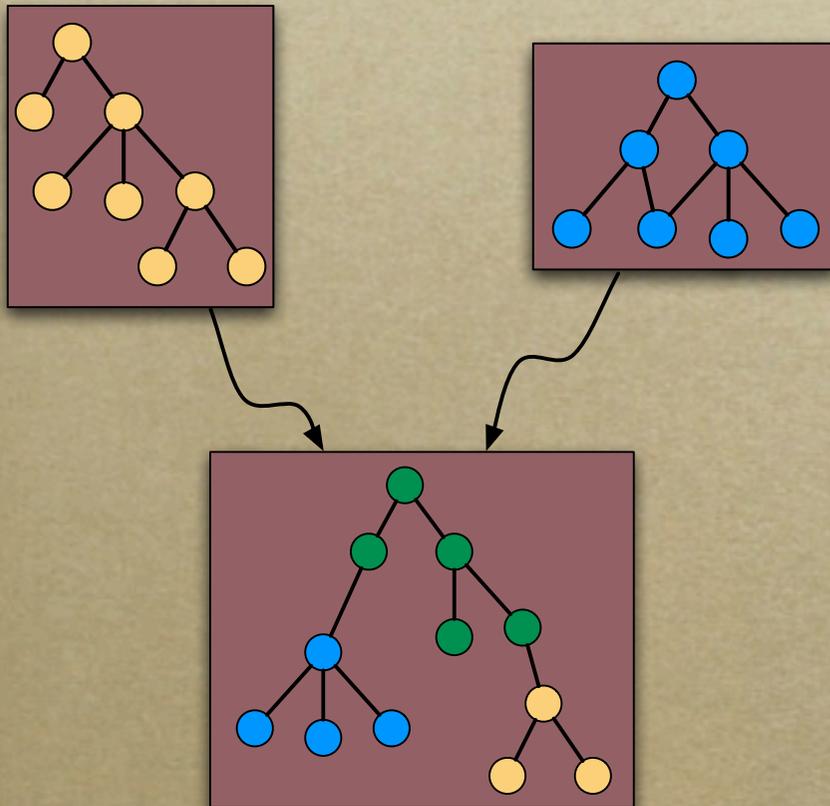
Mapping and Merging



- *Existing ontologies*
 - *cover overlapping domains*
 - *use the same terms with different meaning*
 - *use different terms for the same concept*
 - *have different definitions for the same concept*

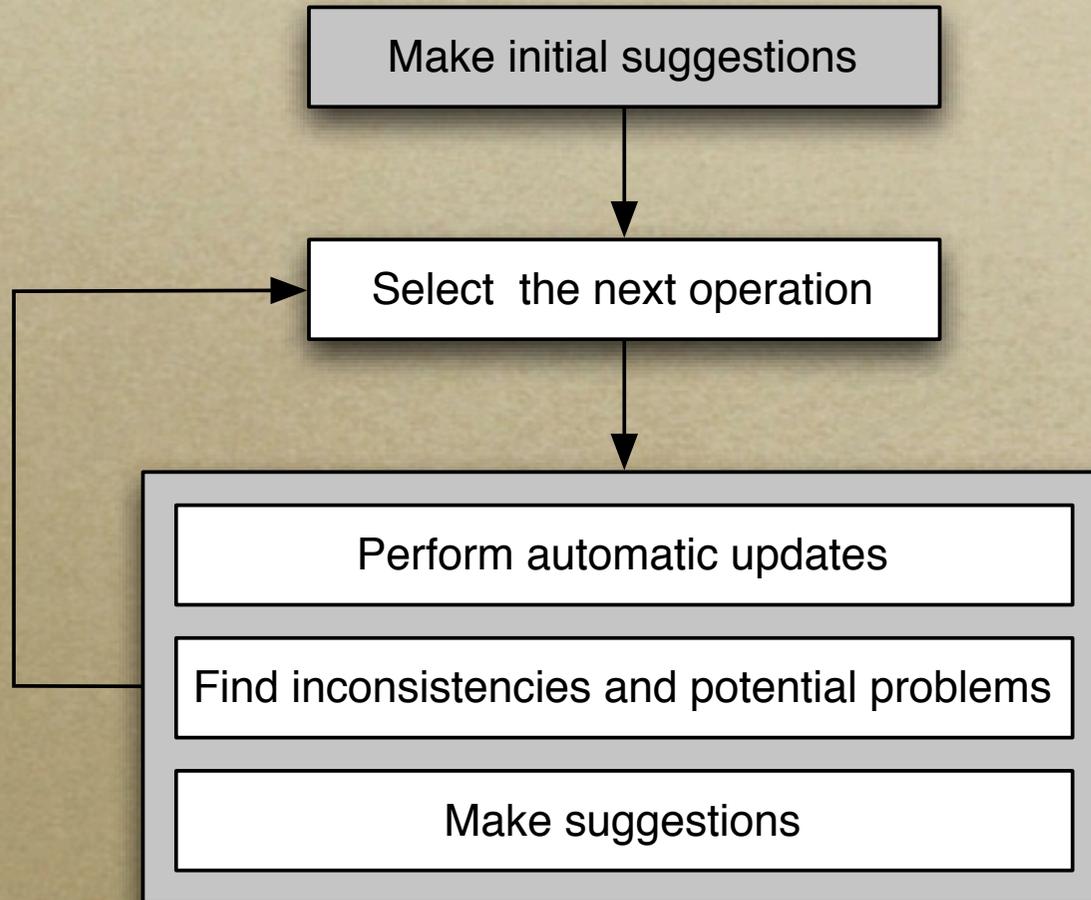


iPrompt: An Interactive Ontology-Merging Tool

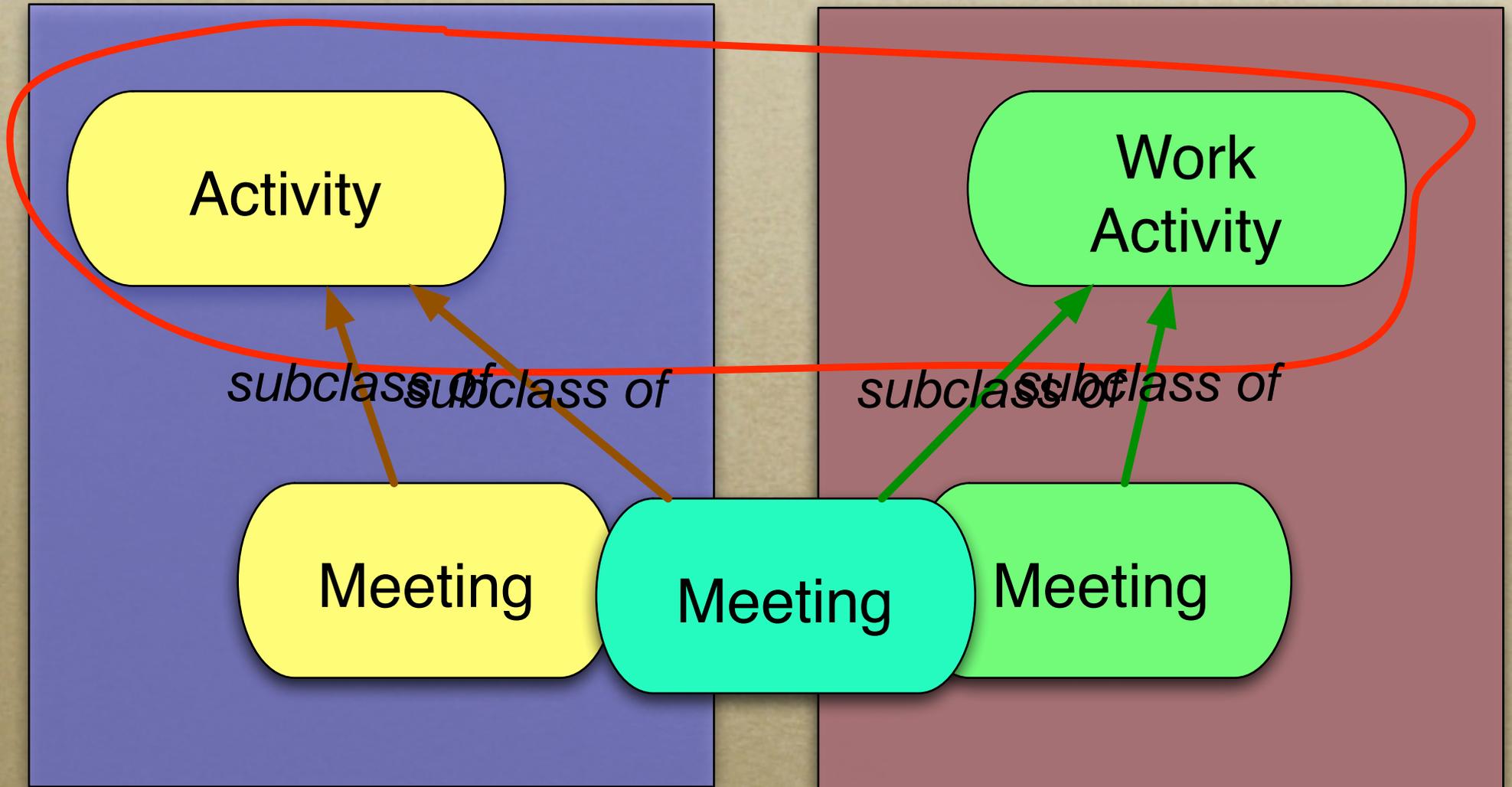


- *iPrompt provides*
 - *Partial automation*
 - *Algorithm based on*
 - concept-representation **structure**
 - **relations** between concepts
 - user's **actions**
- *iPrompt does not provide*
 - *complete automation*

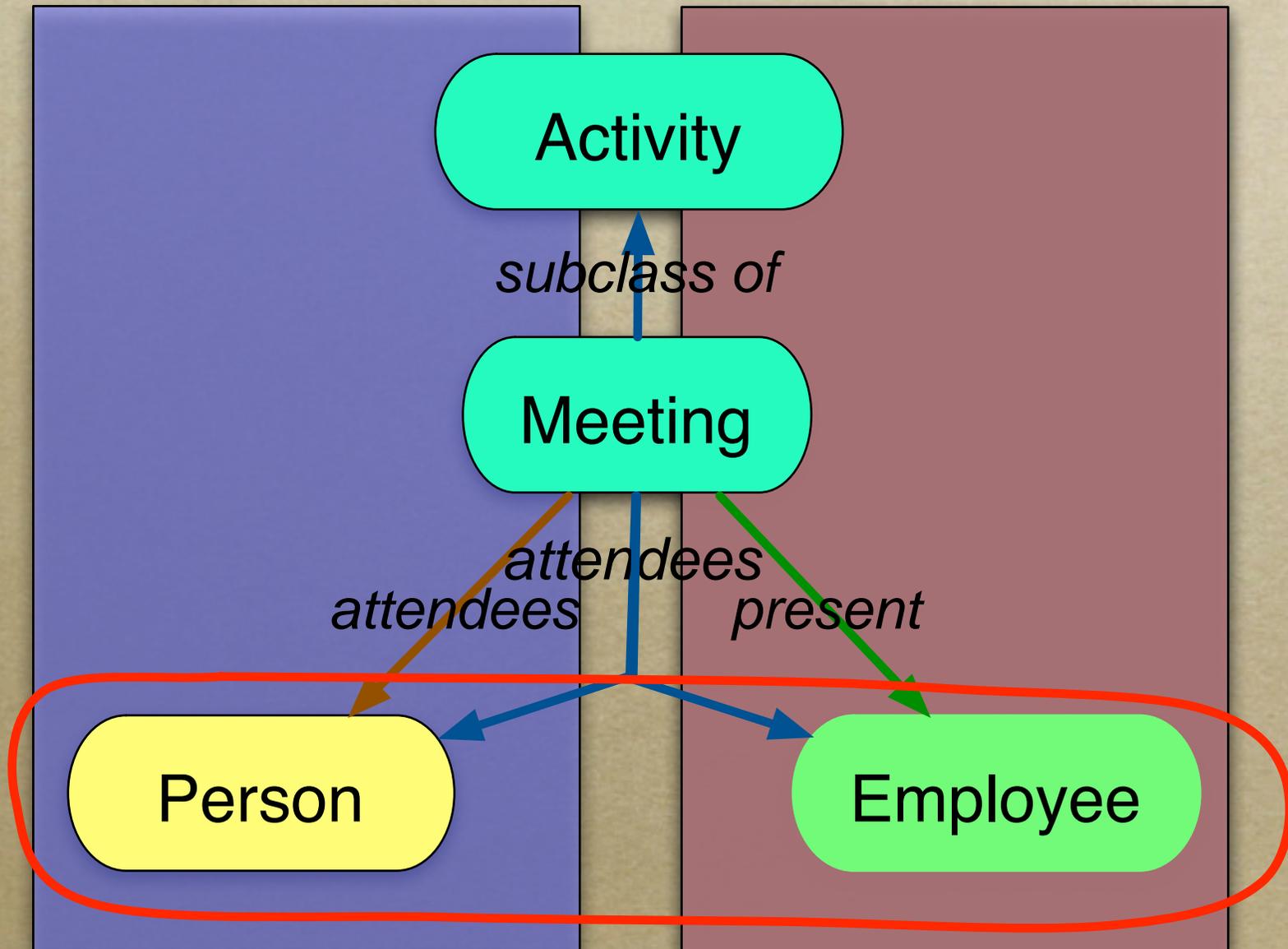
iPrompt Algorithm



Example: Merge Classes



Example: Merge Classes (II)



iPrompt: Initial Suggestions

The screenshot displays the iPrompt interface with several components:

- Top Navigation:** A row of buttons for 'Classes' (yellow circle), 'Slots' (blue square), 'Forms' (green square), 'Instances' (purple diamond), 'Queries' (orange triangle), and 'Prompt' (white button).
- Sub-panels:** Below the navigation are three tabs: 'Suggestions' (selected), 'Conflicts', and 'New operations'.
- To Do list:** A table with columns 'Name', 'Arg1', 'Arg2', and 'Param'. It lists 15 merge operations between classes from 'cmu' and 'umd' domains.
- Result classes:** A panel on the right titled 'Result classes' with a 'current' section containing a tree view of class suggestions.
- Buttons:** A 'Do It' button with a green checkmark is located at the bottom of the 'To Do list' panel.

Name	Arg1	Arg2	Param
merge	● Publication cmu	● Publication umd	
merge	● Employee cmu	● Employee umd	
merge	● Proceedings cmu	● Proceedings umd	
merge	● Organisation cmu	● Organization umd	
merge	● Research_Group cm	● ResearchGroup umd	
merge	● EMail cmu	● Email umd	
merge	● Article cmu	● Article umd	
merge	● Book cmu	● Book umd	
merge	● Thesis cmu	● Thesis umd	
merge	● MastersThesis cmu	● MastersThesis umd	
merge	● PhdThesis cmu	● Thesis umd	
merge	● Person cmu	● Person umd	
merge	● Meeting cmu	● Thing umd	
merge	● Director cmu	● Director umd	

● Classes
■ Slots
■ Forms
◆ Instances
▲ Queries
Prompt

Suggestions
Conflicts
New operations
Result classes

To Do list

Name	Arg1	Arg2	Param
copy	● Image	umd	
merge	■ age--cmu	■ age--umd	
merge	■ sex--cmu	■ sex--umd	
copy			
copy			
mer			
copy			
copy			
copy			
mer			
Rea			
fram			
both			

current

- :THING
- ▶ ● :SYSTEM-CLASS
- Person

Person (instance of :STANDARD-CLASS)

 Concrete ●

Template Slots

Name	Cardinal	Type	Other Fac
■ age--cmu	multiple	Integer	
■ age--umd	multiple	String	
■ birthDate--umd	multiple	String	
■ child--umd	multiple	Instance of Person	
■ doctoralDegreeFr...	multiple	Instance of University--umd_temp	

✓ Do It

After a User Performs an Operation

- *For each operation*
 - *perform* the operation
 - consider possible *conflicts*
 - *identify conflicts*
 - *propose solutions*
 - *analyze* local context
 - create *new suggestions*
 - *reinforce or downgrade* existing suggestions

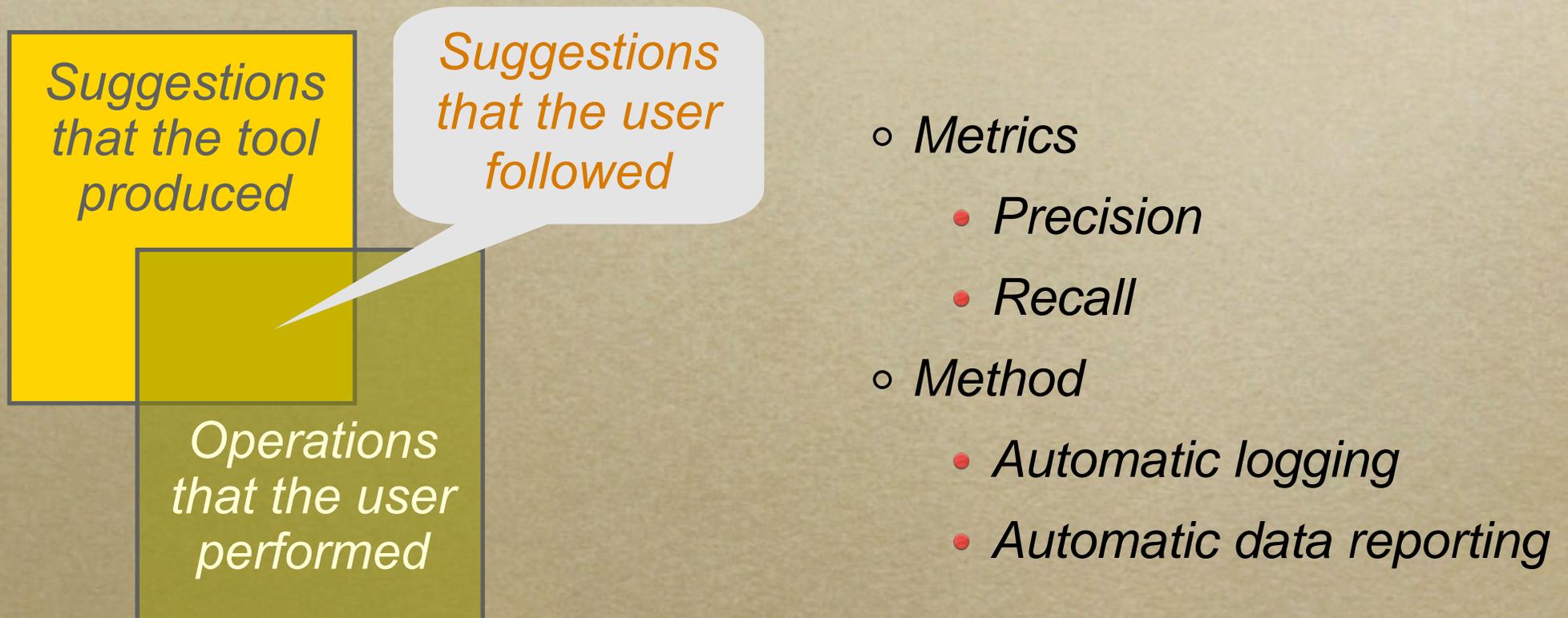
Conflicts

- *Conflicts that PROMPT identifies*
 - *name conflicts*
 - *dangling references*
 - *redundancy in a class hierarchy*
 - *slot-value restrictions that violate class inheritance*

Analyzing Ontology Structure

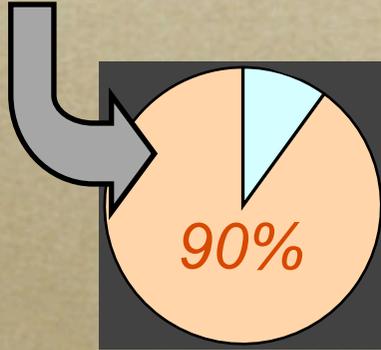
- *Structures that Prompt analyzes*
 - *classes that have the same sets of slots*
 - *classes that refer to the same set of classes*
 - *slots that are attached to the same classes*
- *Local context*
 - *incremental analysis*
 - *consider only the concepts that were affected by the last operation*

Evaluate the Quality of iPROMPT's Suggestions

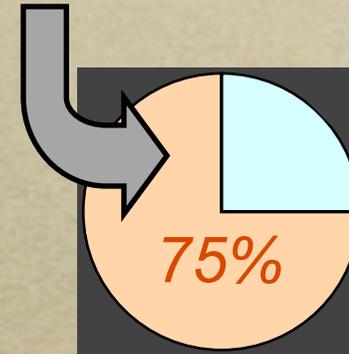


Results: the Quality of iPROMPT's Suggestions

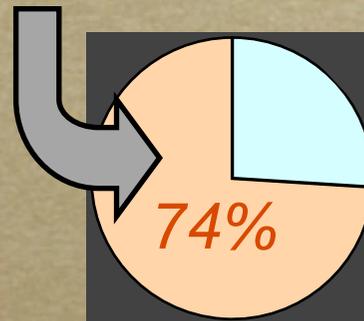
Suggestions that users followed



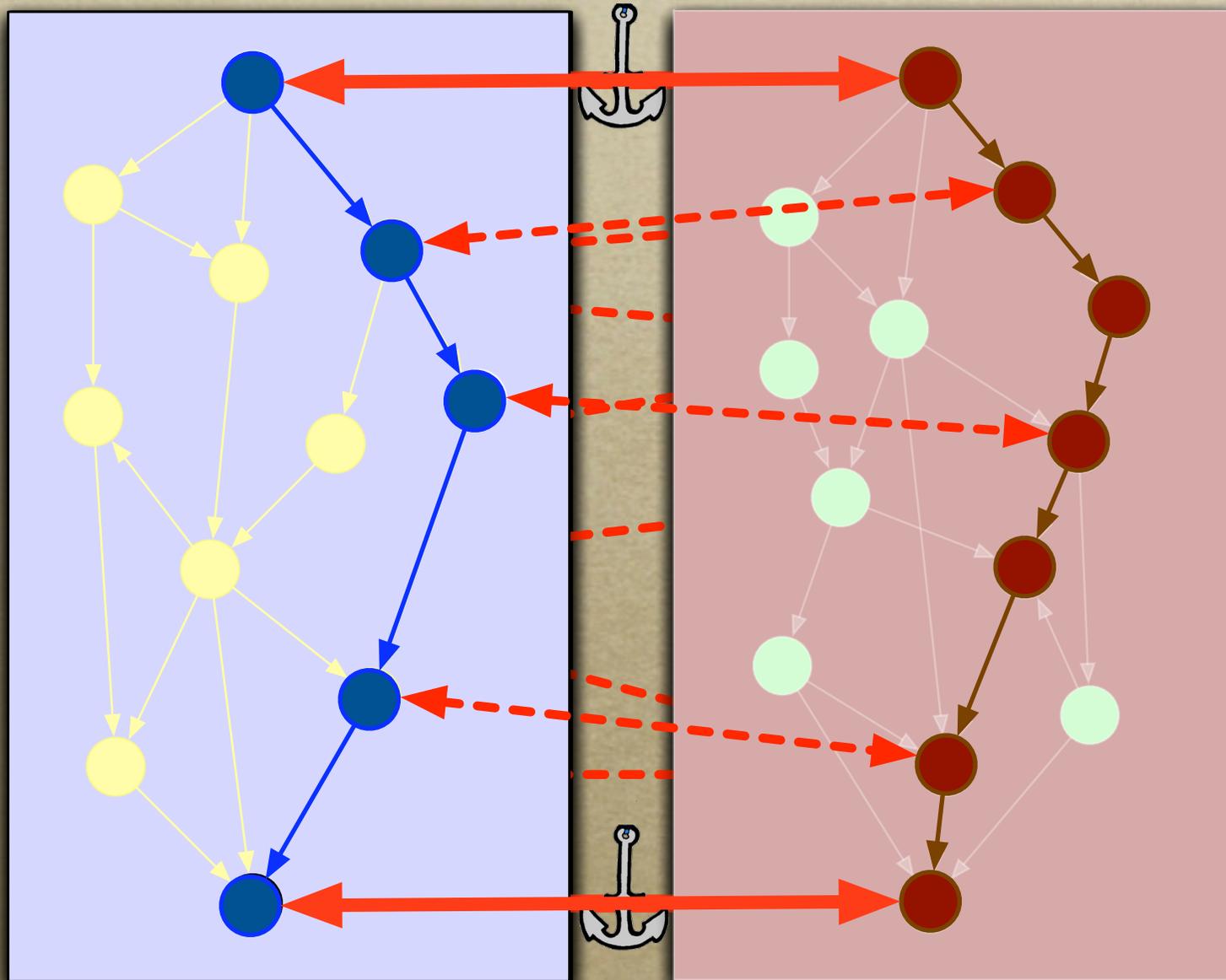
Conflict-resolution strategies that users followed



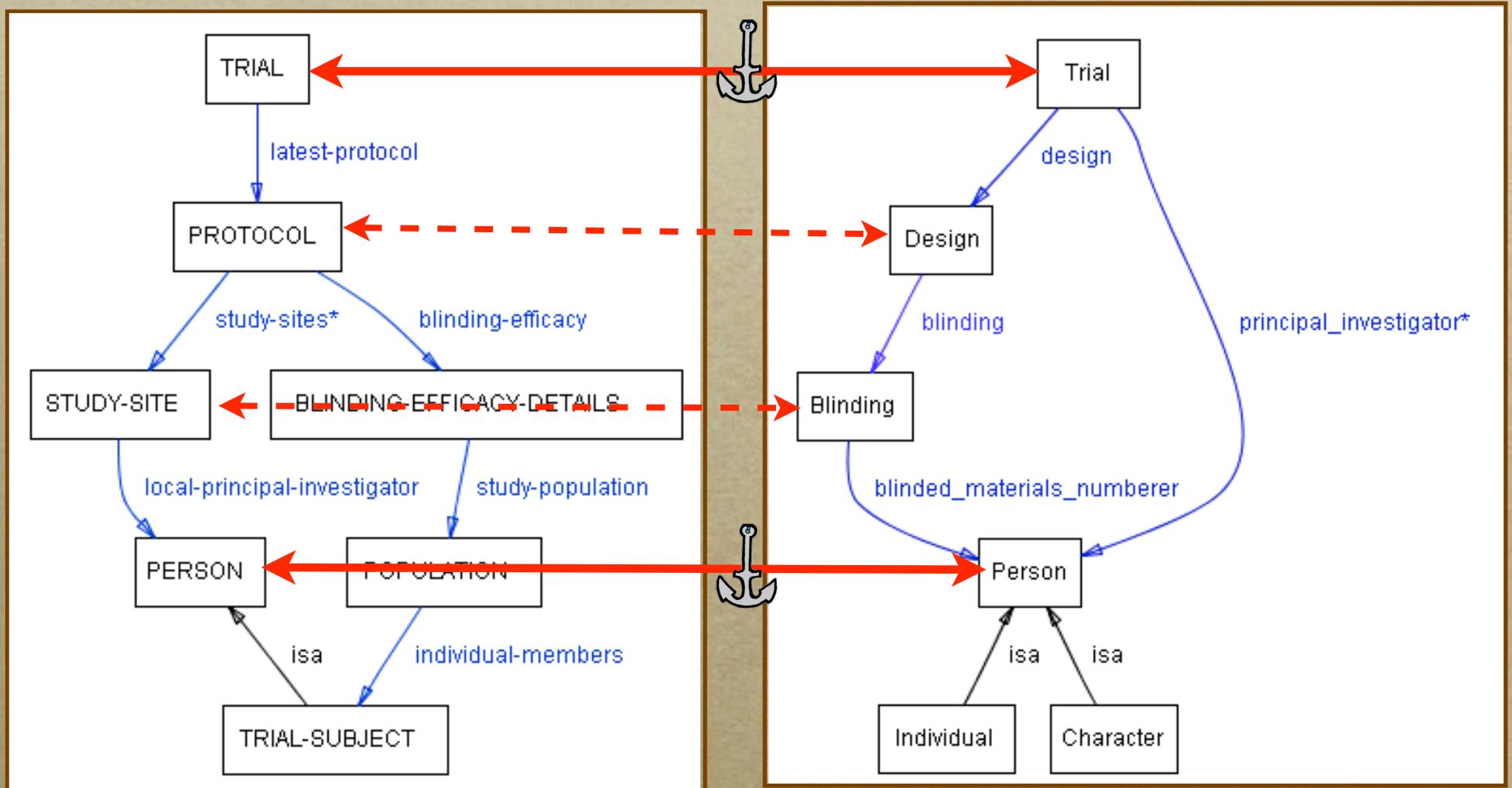
Knowledge-base operations generated automatically



AnchorPrompt: Analyzing Graph Structure



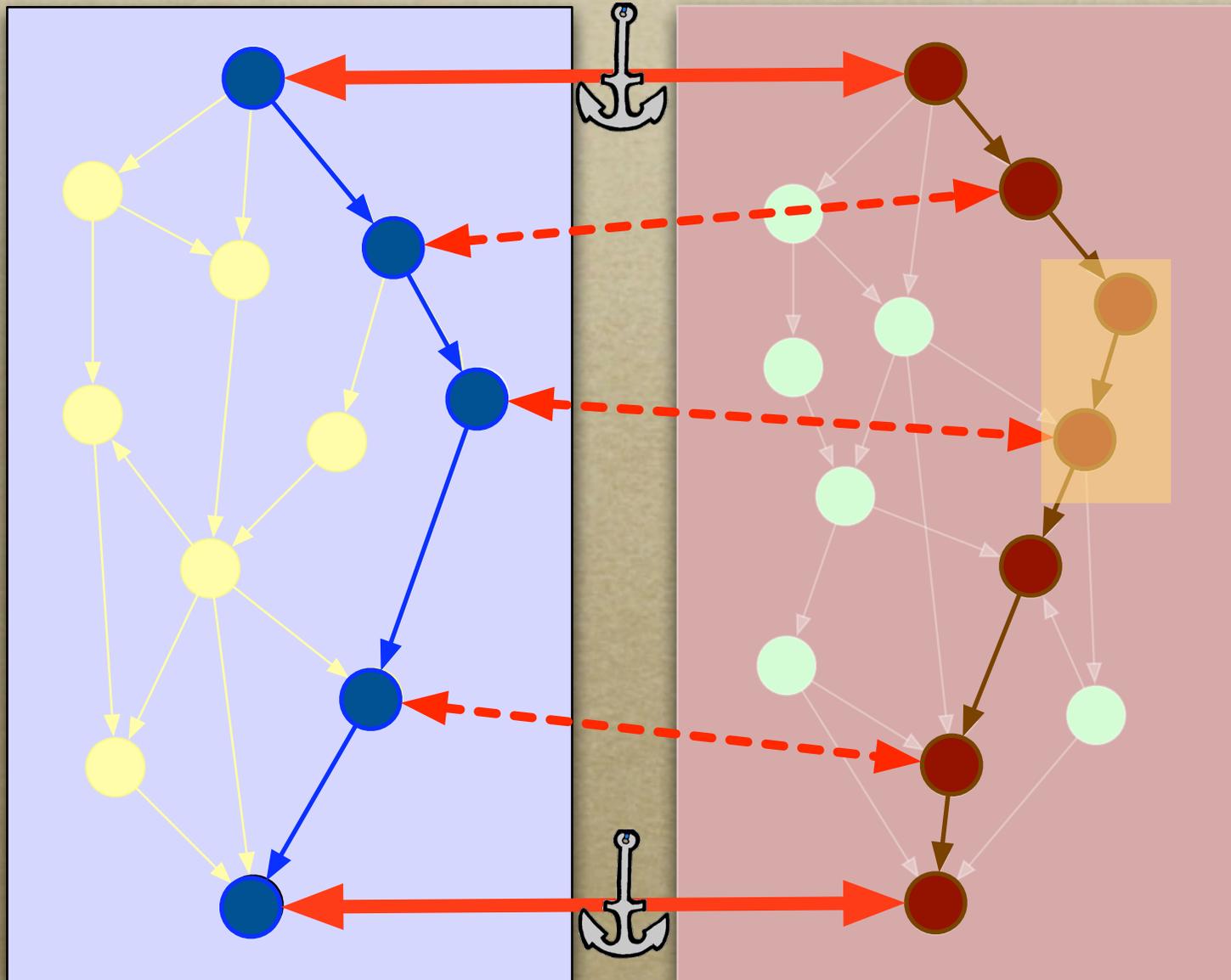
AnchorPrompt: Example



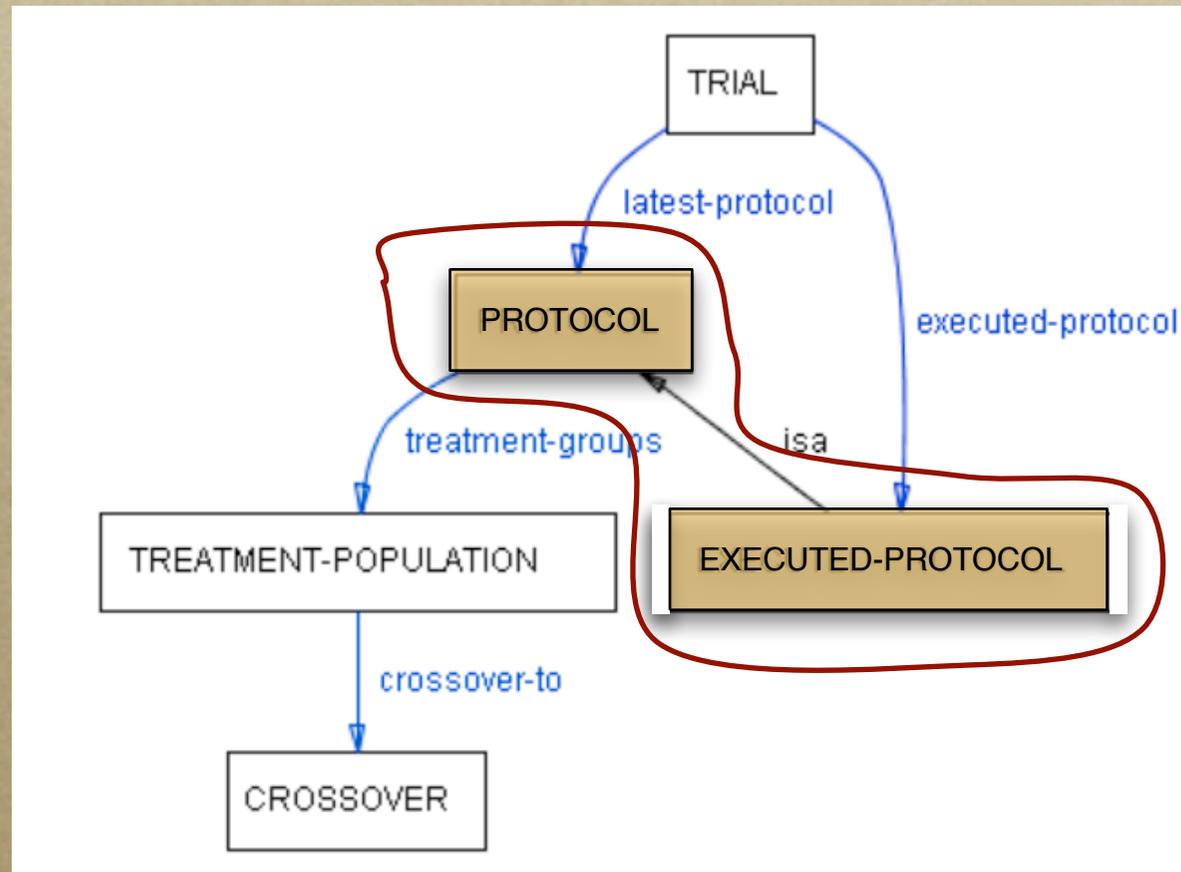
Similarity Score

- *Generate a set of all paths (of length $< L$)*
- *Generate a set of all possible pairs of paths of equal length*
- *For each pair of paths and for each pair of nodes in the identical positions in the paths, increment the similarity score*
- *Combine the similarity score for all the paths*

Equivalence Groups



Equivalence Groups: Example



AnchorPrompt: Example

TRIAL	Trial
PERSON	Person
CROSSOVER	Crossover



PROTOCOL	Design
TRIAL-SUBJECT	Person
INVESTIGATORS	Person
POPULATION	Action_Spec
PERSON	Character
TREATMENT-POPULATION	Crossover_arm

Anchor-PROMPT Evaluation

- *Experiment setup*
 - *Two ontologies from the DAML ontology library describing universities and organizations*
 - *Varying parameters*
 - *maximum path length*
 - *number of anchor pairs*



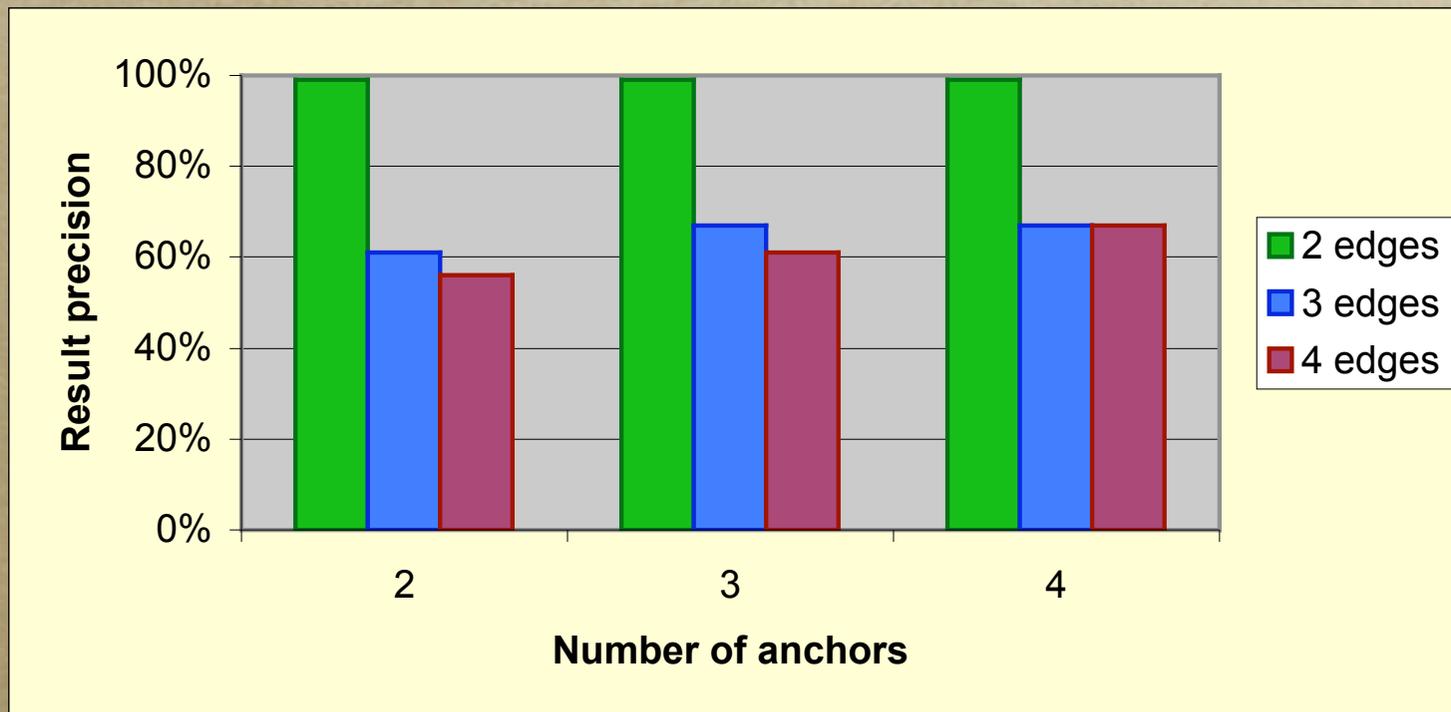
*University of Maryland
research ontology*



CMU Atlas ontology

Anchor-PROMPT: Evaluation Results

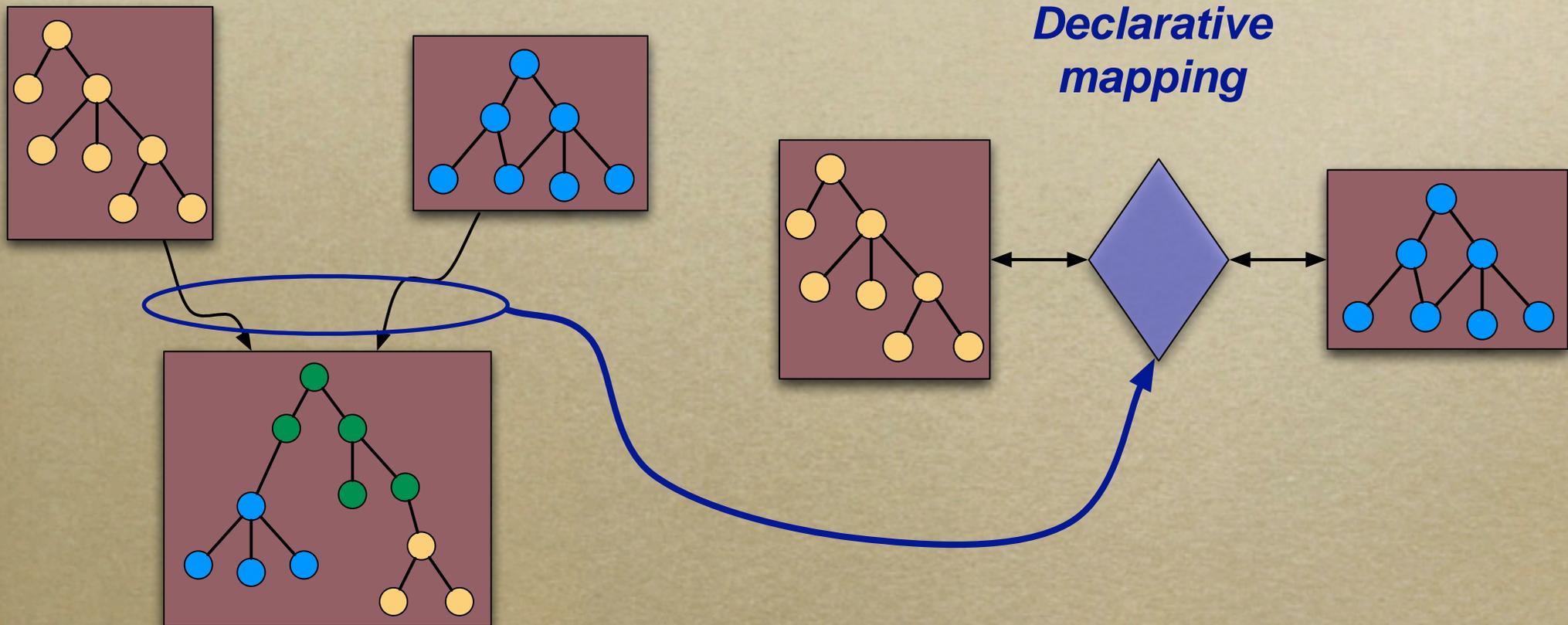
- *Ratio of correct results above the median similarity score*



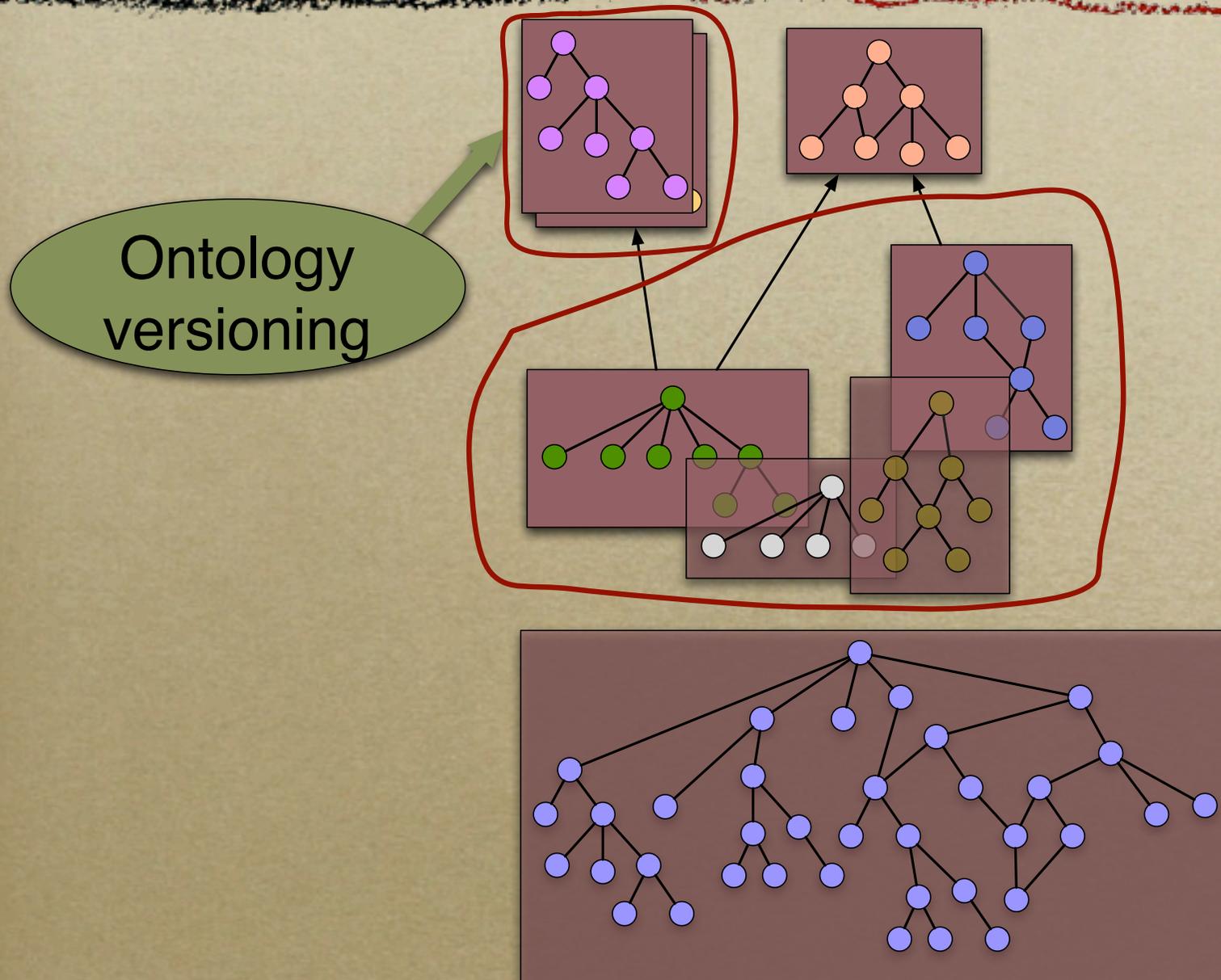
AnchorPrompt Discussion

- *Relies on a **limited** input from the user*
 - *3 anchors → 2-3 new pairs (above median)*
 - *4 anchors → 3 new pairs (above median)*
- *Has **limitations***
 - *source ontologies with very different structure and level of generality*

Combining Merging and Mapping



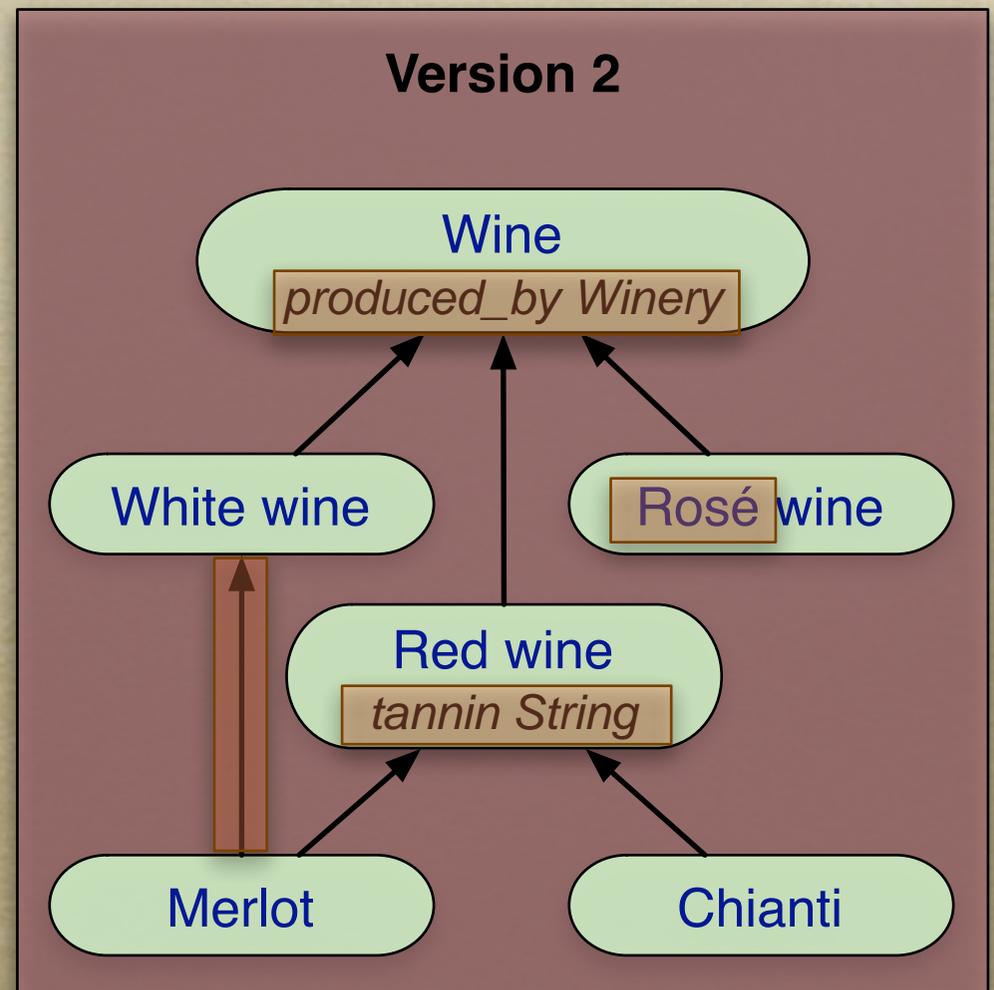
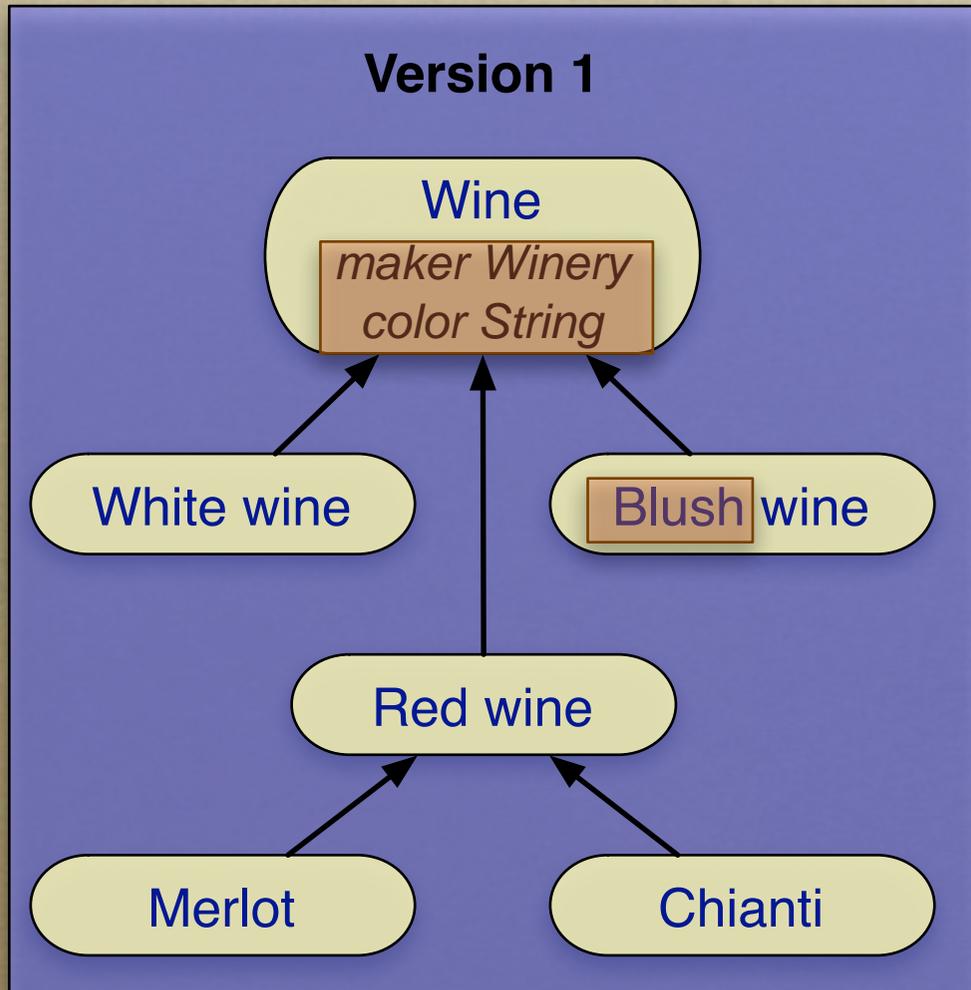
The Messy Picture



Ontology Versioning

- *Ontology development became a **dynamic, collaborative** process*
 - *Need to maintain different ontology versions*
- *CVS-type systems*
 - ✓ *Repository of versions*
 - ✓ *Check-in/check-out mechanisms*
 - ✗ *Version comparison (**diff**)*

Structural Diff

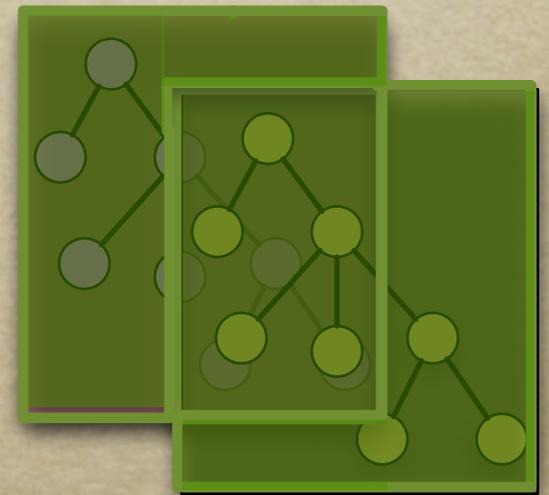


Structural Diff (II)

f1	f2	renamed	operation	map level
	S tannin_level	No	Add	
S color		No	Delete	
C Blush wine	C Rosé wine	Yes	Map	Directly-changed
S maker	S produced_by	Yes	Map	Directly-changed
C Red wine	C Red wine	No	Map	Directly-changed
C Chardonnay	C Chardonnay	No	Map	Changed
C Merlot	C Merlot	No	Map	Changed
C White wine	C White wine	No	Map	Changed
C Wine	C Wine	No	Map	Changed
C Winery	C Winery	No	Map	Unchanged

General Problem: Ontology Matching

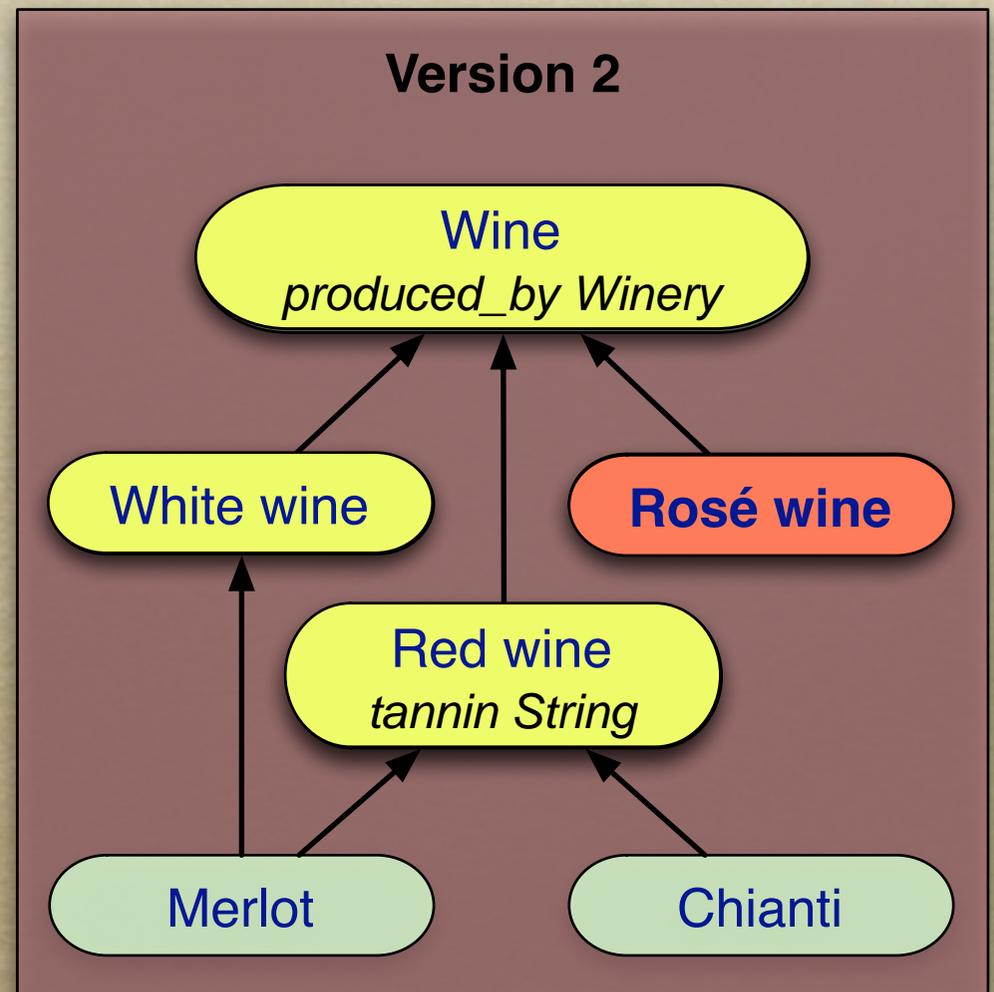
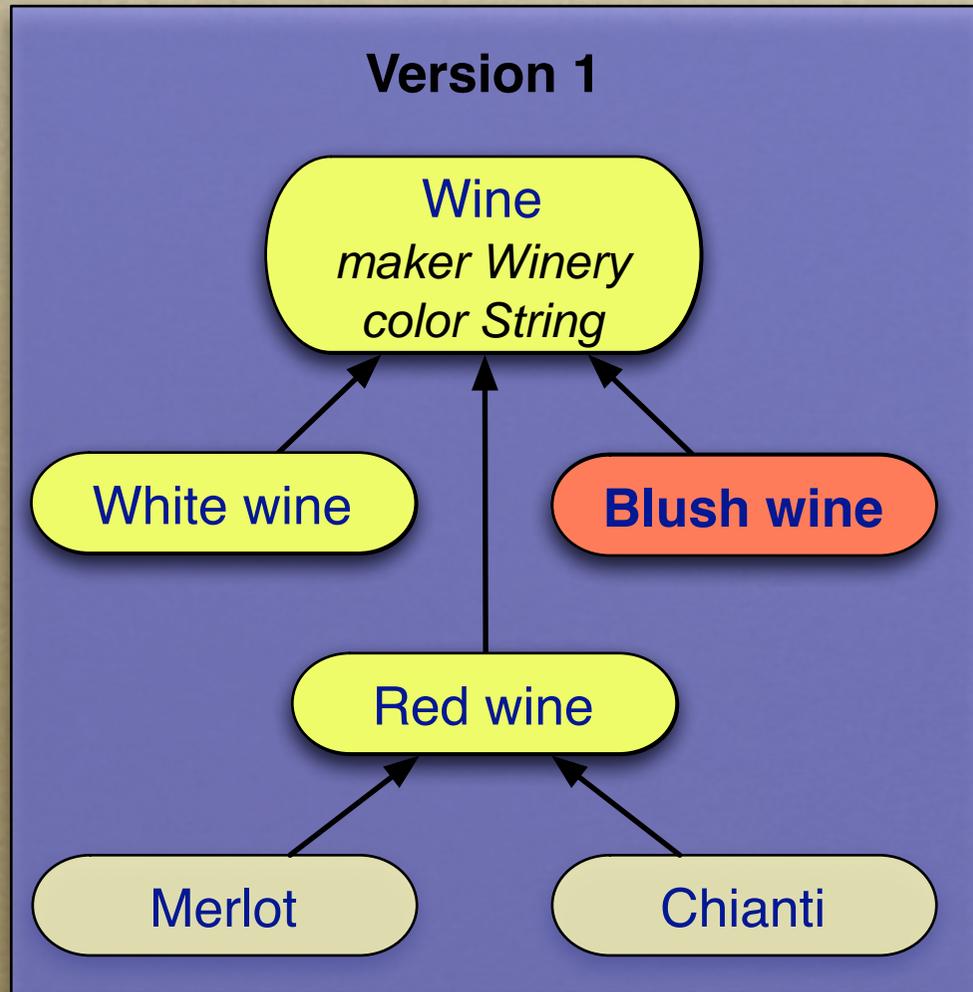
- *Compare ontologies*
- *Find similarities and differences*
 - *Merging: similarities*
 - *Mapping: similarities*
 - *Versioning: differences*
- *Ontology Versioning*
 - *If things look similar, they probably are*
 - *A large fraction of ontologies remains unchanged from version to version*



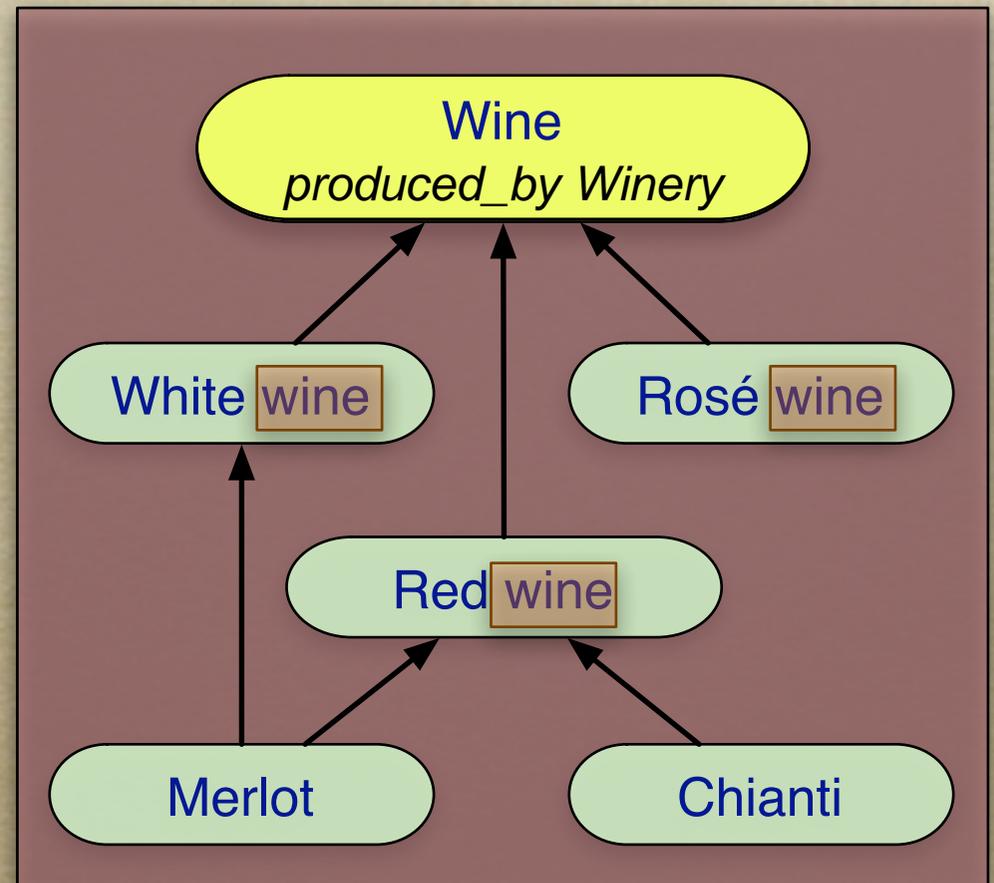
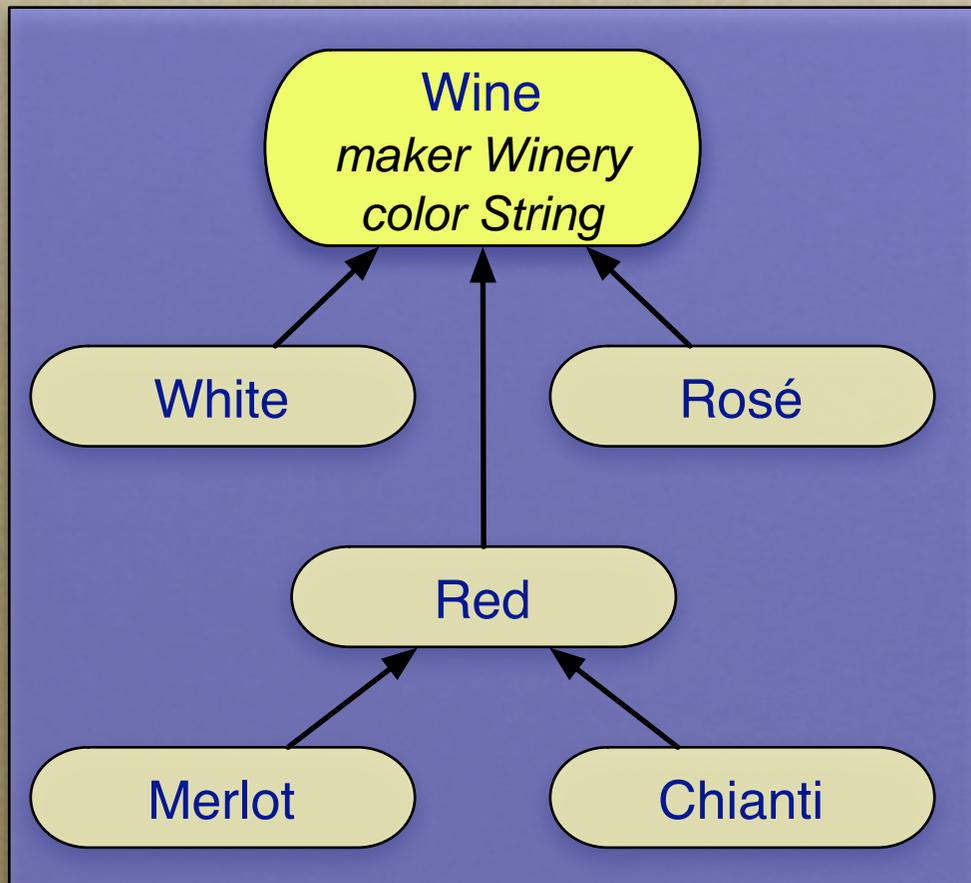
The PrompDiff Algorithm

- *Goal: Find a diff automatically*
- *Consists of two parts*
 - *A set of heuristic matchers*
 - *A fixed-point algorithm to combine the results of the matchers*
- *Can be extended with any number of matchers*

Single Unmatched Siblings



Siblings with the Same Suffixes or Prefixes



Other Matchers

- *Unmatched superclasses*
- *Inverse slots*
- *Multiple unmatched siblings*
- *Instances of the same class with the same slot values*
- *OWL Anonymous classes*

PromptDiff Evaluation

- *Use ontology versions from projects at Stanford Medical Informatics*
 - *EON (~300 frames)*
 - *PharmGKB (~1900 frames)*
- *Both projects*
 - *are collaborative*
 - *use ontologies heavily*
 - *maintain a record of their versions*

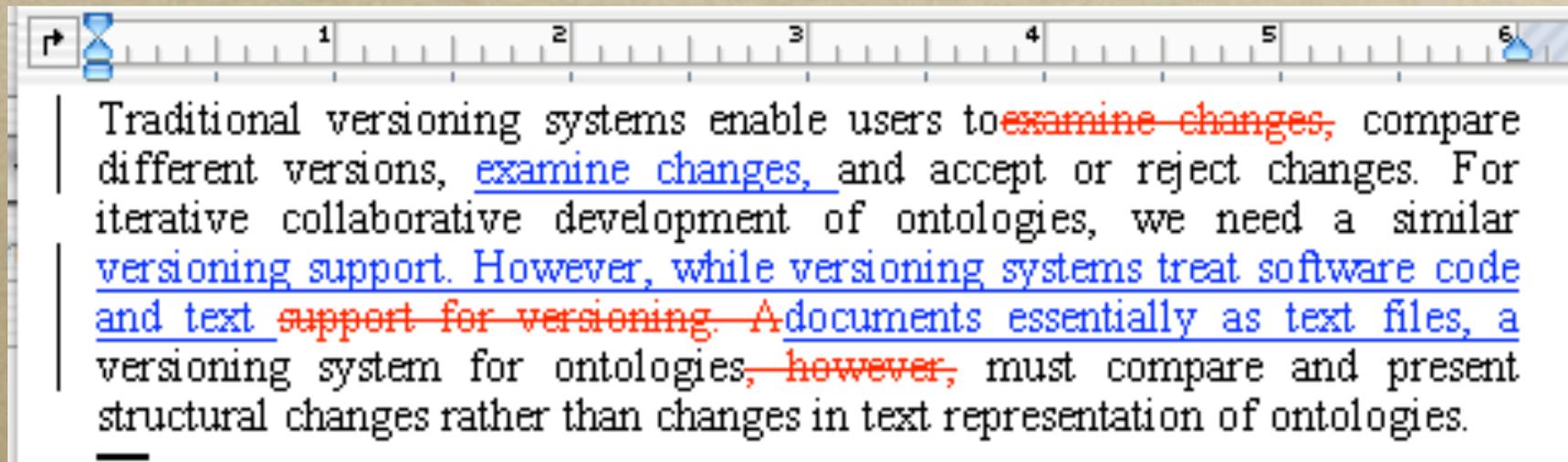
PromptDiff Evaluation

- *We compared results that PromptDiff produced with manually produced results*
- *On average, 98.6% of frames have not changed*
- *We need to consider the accuracy for the remaining 1.4% of frames*

Evaluation Results

- *All frames that PromptDiff matched, it matched correctly*
- *Transformations (match, add, delete) found (recall): 96%*
- *Number of correct transformations (precision): 93%*

Presenting Ontology Diff



Traditional versioning systems enable users to ~~examine changes~~, compare different versions, examine changes, and accept or reject changes. For iterative collaborative development of ontologies, we need a similar versioning support. However, while versioning systems treat software code and text ~~support for versioning~~. A documents essentially as text files, a versioning system for ontologies, ~~however~~, must compare and present structural changes rather than changes in text representation of ontologies.

PromptDiff Interface

The screenshot displays the PromptDiff interface, which is used for managing and comparing classes in a system. The interface is divided into several sections:

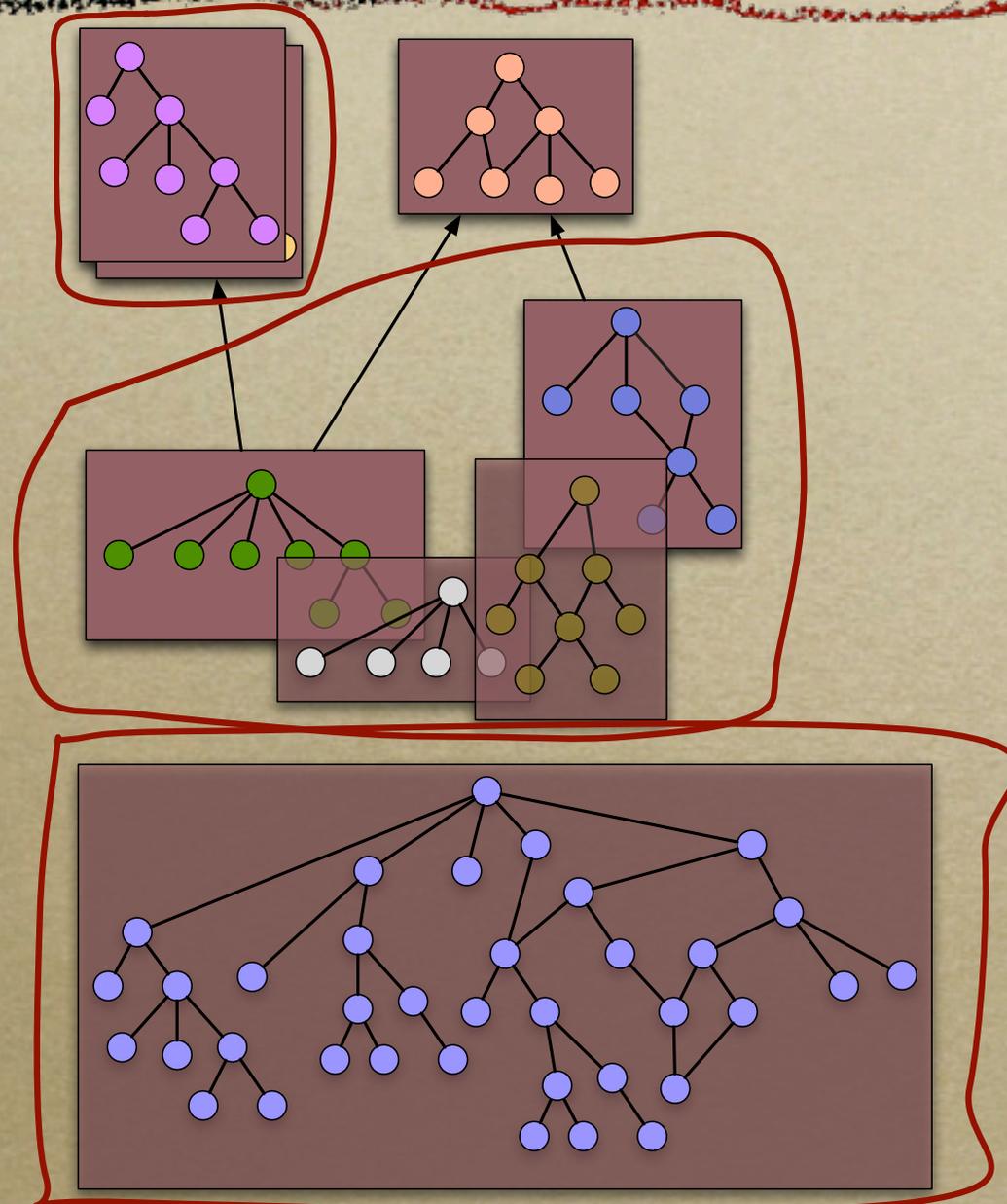
- Navigation:** At the top, there are tabs for "Table view" and "Tree view".
- Class Hierarchy (Left Panel):** A tree view showing the structure of classes. The root is ":THING", followed by ":SYSTEM-CLASS", and then "Wine". Under "Wine", there are sub-classes: "Red wine", "White wine", "Chardonnay", "Riesling", "Dry Riesling", "Sweet Riesling", "Rosé wine", "Winery", and "Meal-course". A tooltip indicates that "class moved from Red wine" for the "Riesling" class.
- Class Details (Right Panel):** A detailed view of the selected "Rosé wine" class (type=:STANDARD-CLASS). It includes:
 - Name:** A text field containing "Rosé wine".
 - Documentation:** A large empty text area.
 - Role:** A dropdown menu currently set to "Concrete".
 - Template Slots:** A table listing slots for the class.
 - Differences:** A table showing changes between the current state and a previous state.
- Superclasses (Bottom Left):** A list of superclasses, currently showing "Wine".

Name	Type	Cardinality	Class
produced_by	Instance	single	cl...

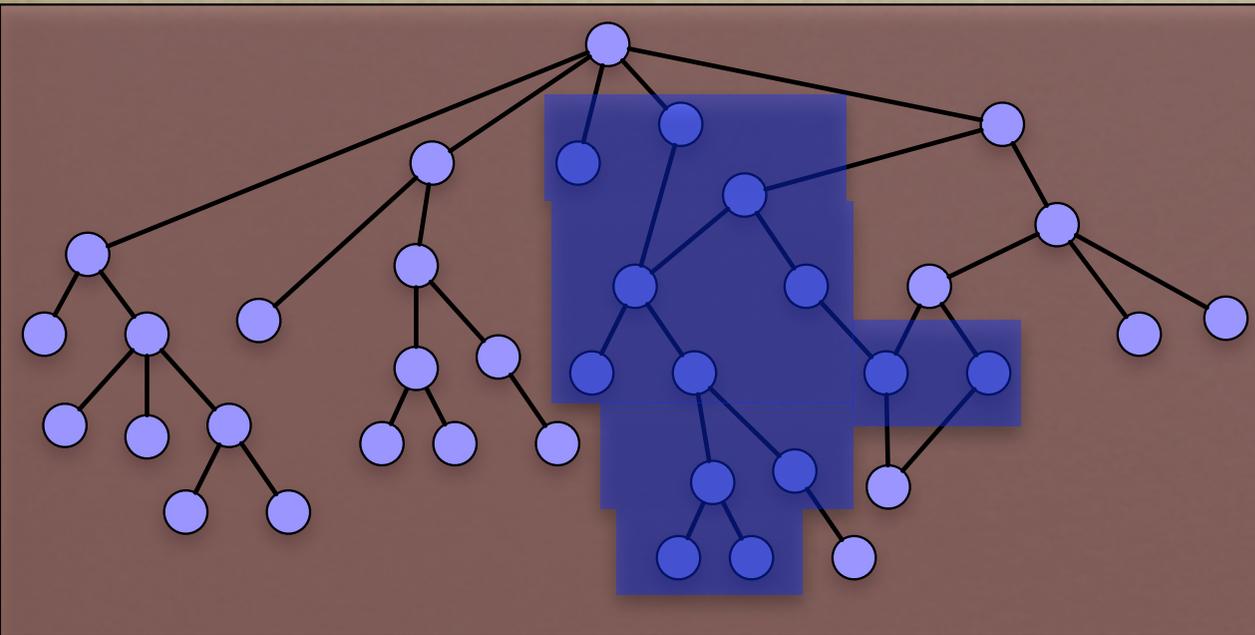
Operation	Slot	Old Value	New Value
own slot value changed	S Name	Blush wine	Rosé wine

Joint work with Michel Klein and Sandhya Kunnatur

The Messy Picture



Ontology Views



- Extract a *self-contained* subset of an ontology
- Ensure that all the *necessary* concepts are defined in the sub-ontology
- Specify the *depth of transitive closure* of relations

Traversal Views

- *Specification of a traversal view*
 - A “*starter*” concept
 - *Relationships* to traverse
 - The *depth* of traversal along each relationship
- *Can find “everything related”*

Defining a View

The image shows a software interface with two main windows. The background window displays a class hierarchy for 'fm_localJuly28'. The hierarchy is as follows:

- fm_localJuly28
 - Cavitated organ
 - Organ with organ
 - Organ with cavita
 - Heart
 - Bone (organ)
 - Long bone
 - Short bone
 - Flat bone
 - Irregular bone

The 'Heart' class is selected. Below the hierarchy, there is a 'copy class' section with a dropdown menu set to 'fm_localJuly28' and a 'Choose class' field containing 'Heart'. There are also checkboxes for 'subclass', 'instance', and 'superclass', with 'superclass' checked. A 'Do It' button is visible at the bottom.

The foreground window is titled 'Slot traversal depth'. It contains the following text: 'Number of levels to copy for specific slots (checking the box and leaving the field blank sets levels to "unlimited")'. Below this text is a list of slots with checkboxes and input fields:

- every slot
- S connected to
- S connecting part
- S connection type
- S constitutional part
- S constitutional part of 4
- S contained in 4
- S contains
- S continuous with

At the bottom of the dialog are 'Select All' and 'Deselect All' buttons, and 'OK' and 'Cancel' buttons at the very bottom.

Saving a View

- *Save* a view as instances in an ontology
- *Replay* the view on a new version
- Determine if a view is “dirty”

Starter Concept

Heart

Subclasses Instances Superclasses

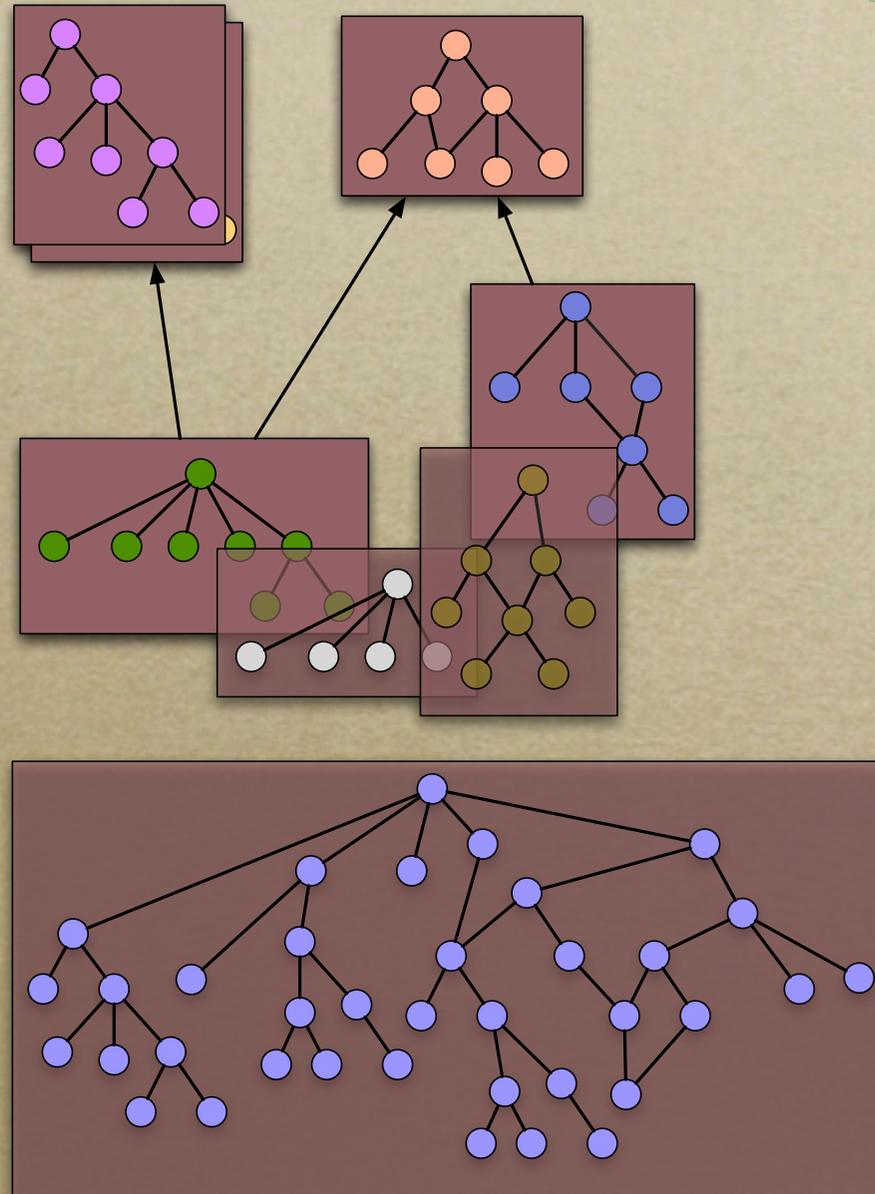
Everything

Number Of Levels

Slot Directives

slot	depth
constitutional part of	4
contained in	4

Dealing with a Messy World



Future Directions

- *Mapping and Merging*
 - *Finding complex mappings*
 - *Dealing with uncertainty*
 - *Maintenance during ontology evolution*
- *Versioning*
 - *Integrating with workflow*
 - *Scalability*
- *Views*
 - *Non-materialized, dynamic views*



"All I'm saying is now is the time to develop the technology to deflect an asteroid"