Ontology matching techniques

Ivo Lasek

Semantic Web

22. 4. 2011

Outline

Outline

Classification along matching dimensions

We distinguish between three primary matching dimensions:

1. The input of the algorithm
2. The characteristics of the matching process
   - Exact algorithms vs. Approximate algorithms
   - Input data interpretation (syntactic, external, semantic)
3. The output of the algorithms
   - The graduation of the output
   - The kind of relations between entities the system is able to provide (equivalence =, subsumption $\sqsubseteq$ and incompatibility $\perp$)

Outline

Similarity

Similarity

A similarity $\sigma : o \times o \rightarrow \mathbb{R}$ is a function from a pair of entities to a real number expressing the similarity between two objects such that:

- $\forall x, y \in o, \sigma(x, y) \geq 0$ *(positiveness)*
- $\forall x \in o, \forall y, z \in o, \sigma(x, x) \geq \sigma(y, z)$ *(maximality)*
- $\forall x, y \in o, \sigma(x, y) = \sigma(y, x)$ *(symmetry)*

- A real number which denotes how similar two entities are
- Greater than 0
- The similarity between two same objects is the highest one
- The similarity between x and y is the same as between y and x

Dissimilarity

Dissimilarity

Given a set o of entities, a dissimilarity $\delta : o \times o \rightarrow \mathbb{R}$ is a function from a pair of entities to a real number such that:

- $\forall x, y \in o, \delta(x, y) \geq 0$ *(positiveness)*
- $\forall x \in o, \delta(x, x) = 0$ *(minimality)*
- $\forall x, y \in o, \delta(x, y) = \delta(y, x)$ *(symmetry)*

- A real number which denotes how different two entities are
- Greater than 0
- Two same objects have the dissimilarity = 0
- The dissimilarity between x and y is the same as between y and x

Normalised (dis)similarity

Normalised (dis)similarity

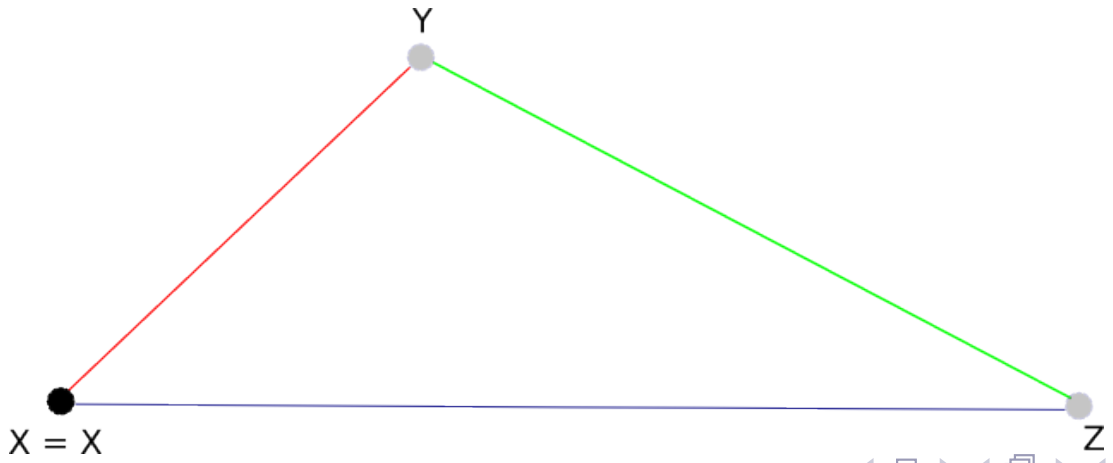
A (dis)similarity is said to be normalised if it ranges over the unit interval of real numbers [0 1]. A normalised version of a (dis)similarity σ (respectively, δ) is denoted as $\bar{\sigma}$ (respectively, $\bar{\delta}$).

Distance

Distance

A distance (or metric) $\delta : o \times o \rightarrow \mathbb{R}$ is a dissimilarity function satisfying the definiteness and triangular inequality:

- $\forall x, y \in o, \delta(x, y) = 0$ if and only if $x = y$ *(definiteness)*
- $\forall x, y, z \in o, \delta(x, y) + \delta(y, z) \geq \delta(x, z)$ *(triangular inequality)*



Ultrametric

Ultrametric

Given a set of entities, an ultrametric is a metric such that:

- $\forall x, y, z \in o, \delta(x, y) \leq max(\delta(x, z), \delta(y, z))$ *(ultrametric inequality)*

- For all x, y, z. At least two of $\delta(x, y)$, $\delta(y, z)$, and $\delta(x, z)$ are the same.

Outline

Element-level techniques

Element-level techniques

Consider ontology entities or their instances in isolation from their relations with other entities or their instances.

Name-based techniques

- Compare strings
- Can be applied to the name, the label or the comments of entities
- Can be used for comparing class names and/or URIs
- The main problem: existence of synonyms and homonyms
  - **Synonyms** are different words used to name the same entity (e.g. Article and Paper)
  - **Homonyms** are words used to name different entities. One word can have multiple senses (peer, walk etc.).
- Another problems: words from different languages, syntactic variations (Compact disc, CD, C.D., CD-ROM etc.).

String-based methods

- Take advantage of the structure of the string
- **Normalisation** can help improve subsequent comparisons
  - Case and blank normalisation
  - Diacritics and digit suppression
  - Link stripping - replacing apostrophes and blank underlines into dashes or blanks
  - Punctuation elimination

Substring techniques

- **String equality** returns 0 if the strings under consideration are not identical and 1 if they are identical
- **Hamming distance** counts the number of positions in which the two strings differ
- **Substring test** considers that strings are very similar when one is a substring of another (only two values 0 and 1)
- **Substring similarity** considers the length of the longest common substring
- ***n*-gram similarity** computes the number of common n-grams (sequence of n characters) between strings. For instance, trigrams for the string article are: __a, _ar, art, rti, tic, icl, cle, le_, e__.

Edit distance

- Edit distance between two objects is the minimal cost of operations to be applied to one of the objects in order to obtain the other one
- Designed for measuring similarity between strings that may contain spelling mistakes
- The considered operations:
  - Insertion of a character
  - Replacement of a character by another
  - Deletion of character
- Each operation is assigned a cost and the distance between two strings is the sum of cost of each operation on the less costly set of operations

Levenshtein

Novel ➙ Pocket

- sub: Novel ➙ Povel
- sub: Povel ➙ Pocel
- ins: Pocel ➙ Pockel
- sub: Pockel ➙ Pocket
- **Distance: 4**

Science ➙ Politics

- ins: Science ➙ PScience
- sub: PScience ➙ Politics
- **Distance: 6**

http://www.merriampark.com/ld.htm

Levenshtein

Novel ➙ Pocket

- sub: Novel ➙ Povel
- sub: Povel ➙ Pocel
- ins: Pocel ➙ Pockel
- sub: Pockel ➙ Pocket
- **Distance: 4**

Science ➙ Politics

- ins: Science ➙ PScience
- sub: PScience ➙ Politics
- Distance: 6

http://www.merriampark.com/ld.htm

Levenshtein

Novel ➙ Pocket

- sub: Novel ➙ Povel
- sub: Povel ➙ Pocel
- ins: Pocel ➙ Pockel
- sub: Pockel ➙ Pocket
- **Distance: 4**

Science ➙ Politics

- ins: Science ➙ PScience
- sub: PScience ➙ Politics
- Distance: 6

http://www.merriampark.com/ld.htm

Levenshtein

Novel ➙ Pocket

- sub: Novel ➙ Povel
- sub: Povel ➙ Pocel
- ins: Pocel ➙ Pockel
- sub: Pockel ➙ Pocket
- **Distance: 4**

Science ➙ Politics

- ins: Science ➙ PScience
- sub: PScience ➙ Politics
- Distance: 6

http://www.merriampark.com/ld.htm

Levenshtein

Novel ➙ Pocket

- sub: Novel ➙ Povel
- sub: Povel ➙ Pocel
- ins: Pocel ➙ Pockel
- sub: Pockel ➙ Pocket
- **Distance: 4**

Science ➙ Politics

- ins: Science ➙ PScience
- sub: PScience ➙ Politics
- Distance: 6

http://www.merriampark.com/ld.htm

Levenshtein

Novel ➙ Pocket

- sub: Novel ➙ Povel
- sub: Povel ➙ Pocel
- ins: Pocel ➙ Pockel
- sub: Pockel ➙ Pocket
- **Distance: 4**

Science ➙ Politics

- ins: Science ➙ PScience
- sub: PScience ➙ Politics
- **Distance: 6**

http://www.merriampark.com/ld.htm

Token-based techniques [1]

- Consider a string as a (multi)set of words (bag of words)
- usually work well on long texts (aggregating different sources of strings: identifiers, labels, comments etc.)
- Splitting strings into tokens
- Many different similarities or dissimilarities being applied to sets of entities can be applied to these bags (e.g. union, intersection, cardinality etc.)
- Usually transformation of bags into vectors (each dimension is a term (token), each position in the vector is the number of occurrences of the token in the corresponding bag of words)
- $\Rightarrow$ Usual metric space distances can be used (e.g. Cosine similarity)

Language-based methods

- Strings are seen as texts which can be segmented into words
- Rely on using Natural Language Processing (NLP) techniques to help extract the meaningful terms from a text
- We distinguish methods which rely on algorithms only (**Intrinsic methods**) and those which make use of external resources (**Extrinsic methods**) such as dictionaries

Intrinsic methods: Linguistic normalisation

- Aims at reducing each form of a term to some standardised form that can be easily recognised
  - **Tokenisation** consists of segmenting strings into sequences of tokens (for example peer-reviewed periodic publication becomes ⟨peer, reviewed, periodic, publication⟩).
  - **Lemmatisation:** The strings underlying tokens are morphologically analysed in order to reduce them to normalised basic forms (suppressing tense, gender or number marks).
  - **Term extraction:** Terminology extractors identify terms from the repetition of morphologically similar phrases in the texts and the use of patterns.
  - **Stopword elimination** Tokens that are recognised as articles, proposition, conjunctions, etc. are marked to be discarded.

Extrinsic methods

- Several kinds of linguistic resources can be exploited in order to find similarities between terms
  - **Lexicons** or dictionaries are sets of words with a natural language definitions of these words
  - **Multi-lingual lexicons:** instead of definition the equivalent terms in another language are listed
  - **Semantico-syntactic lexicons** record names, their categories and the types of arguments taken by verbs and adjectives (e.g. to flow takes a liquid as subject and has no object)
  - **Thesauri:** A thesaurus is a kind of lexicon to which some relational information has been added - usually contains hypernyms (synonyms)
  - **Terminologies** Thesauri which contain phrases rather than single words.

Measures based on linguistic resources

- **Synonymy similarity:** 1 if the words are synonyms, 0 otherwise
- **Cosynonymy similarity** considers the count of common synonyms between two terms
- Some measures are based on the assumption that the more probable a concept, the less information it carries (for example **Resnik semantic similarity** - each synset is associated with a probability of occurrence)
- **Gloss overlap:** Use of definitions (glosses) given to terms - comparing the glosses (i.e. count of common terms between glosses)

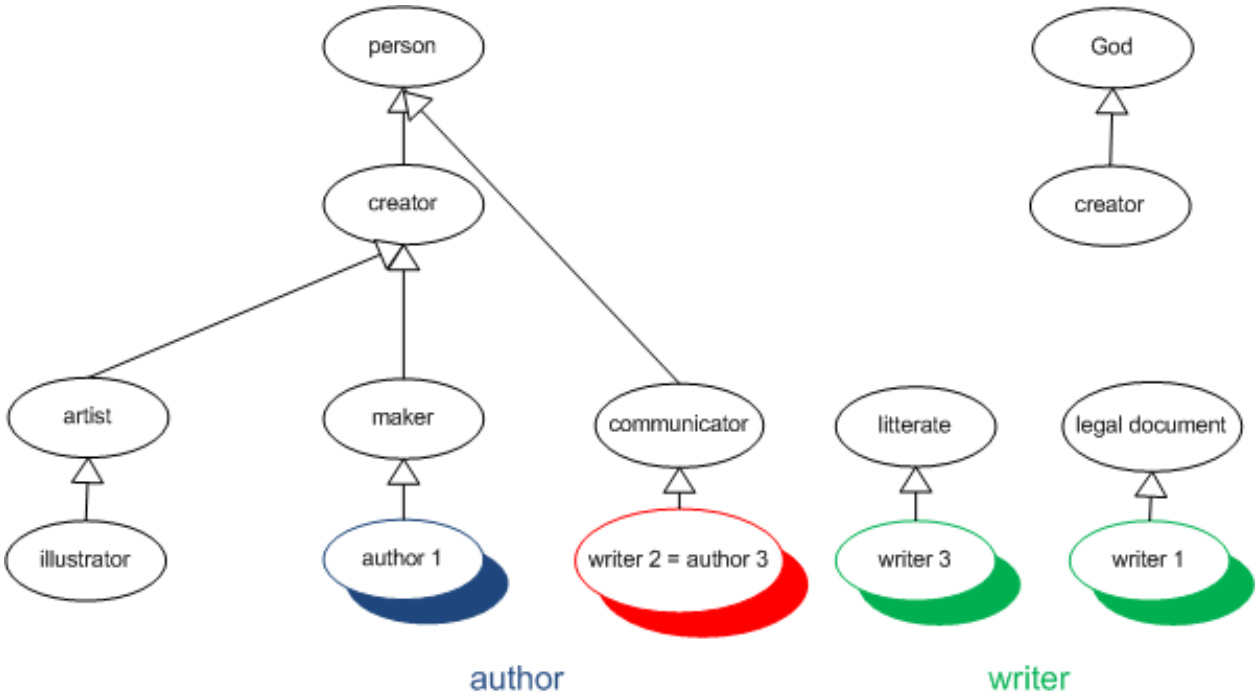
Cosynonymy simlarity

Cosynonymy simlarity

Given two terms s and t and a synonymy resource Σ, the cosynonymy similarity is a similarity $\sigma : \mathbb{S} \times \mathbb{S} \rightarrow [01]$ such that:

$$\sigma(s,t) = \frac{|\Sigma(s) \cap \Sigma(t)|}{|\Sigma(s) \cup \Sigma(t)|}$$

Cosynonymy simlarity



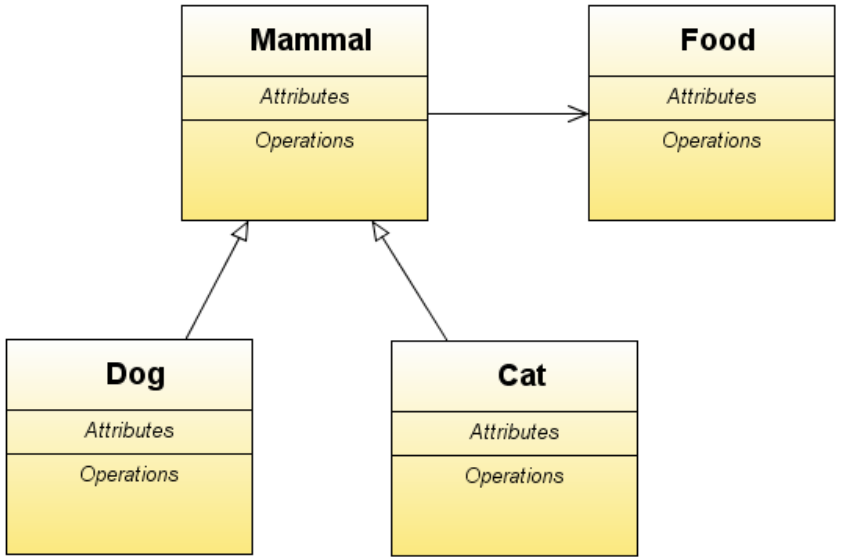
Cosynonymy similarity

Cosynonymy similarity is $1/4 = 0.25$

Structure-level techniques

Structure-level techniques

Consider the ontology entities or their instances to compare their relations with other entities or their instances.



Internal structure based methods

- Methods based on the internal structure of entities - use such criteria as the set of their properties (attributes and relations), their cardinality or multiplicity, and the transitivity or symmetry of their properties to calculate the similarity between them
- Commonly used to create correspondence cluster rather than to discover accurate correspondence between entities

Internal structure comparisons

- **Property comparison and keys:** Two classes identified in the same way are likely to represent the same set of objects.
- **Data type comparison:** Ideally, the proximity between datatypes should be maximal when these are the same types, lower when the types are compatible (i.e. integer and float) and the lowest when they are non compatible.
- **Domain comparison:** The domain contrary to datatype is just a subset of a particular datatype (like [10 12]). Based on the intuition that the distance between domains depends on the difference between the values they cover in isolation and in common.
- **Comparing multiplicities and properties:** Comparing classes based on comparison of compatibility between cardinalities (like by data type comparison).

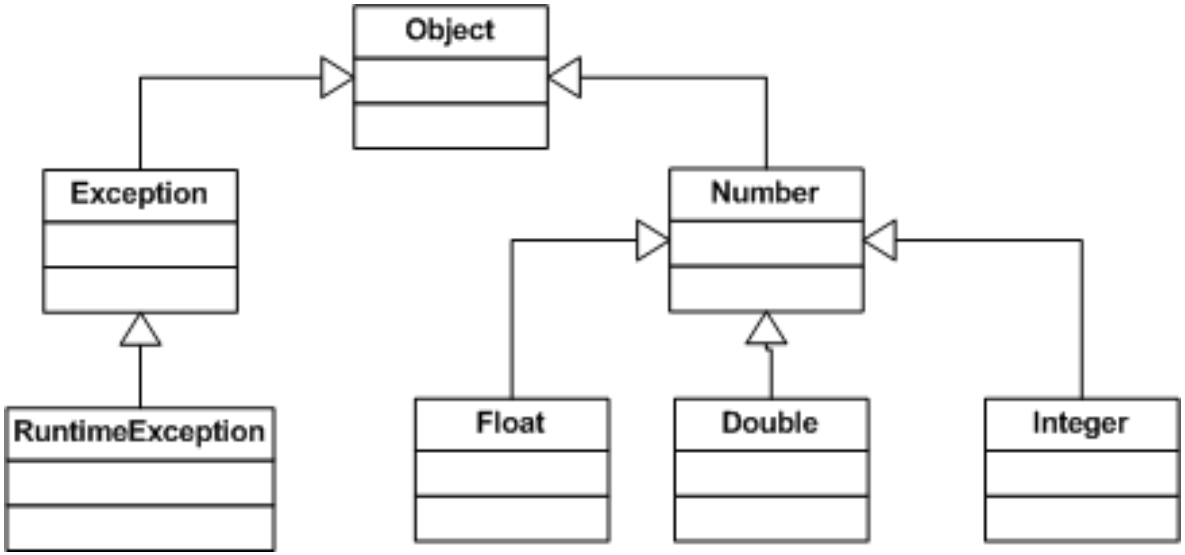
Relational structure

- An ontology can be considered to be a graph whose edges are labelled by relation names
- Finding the correspondence between elements of such graphs corresponds to solving form of the graph homomorphism problem
- It can be related to finding maximum common directed subgraph
- In ontology matching, the problem is encoded as an optimisation problem $\Rightarrow$ the subgraphs do not have to be maximal

Relation types

- There are three types of relations that have been considered so far in relational structure:
  1. **Taxonomic structure:** The graph is made with the *subClassOf relations*
     - counting the number of edges in the taxonomy between two classes
     - some measures count in the distance from the root as well
     - counting the number of common superclasses
  2. **Mereologic structure:** The structure corresponding to a *part-of* relationship - the classes are considered more similar if they share similar parts
  3. **Relations:** The general problem of matching entities based on all their relations. But the relation graph can contain circles.

Structural topological dissimilarity on hierarchies



- Counting the intermediate edges
- $\delta(\mathit{Float}, \mathit{Double}) = 2$, $\delta(\mathit{Float}, \mathit{Number}) = 1$
- But $\delta(\mathit{Number}, \mathit{Exception}) = 2$. Really?
- Is Exception and Number as similar as Float and Double?

Extensional techniques

Extensional techniques

- When two ontologies share the same set of individuals
- **Presumption:** When two classes share the same set of individuals, then these classes represent probably a correct match

Common extension comparison

- The easiest way to compare classes is to compare the intersection of their instances
- The problem is that only small amounts of incorrect data may lead the system to draw wrong conclusion on domain relationships
- A way to refine this is to use *Hamming distance* between two extensions
- It is also possible to compute the similarity based on the probabilistic properties (*Jaccard similarity*)

Hamming distance

Hamming distance

The Hamming distance between two sets is a similarity function $\delta : 2^E \times 2^E \rightarrow \mathbb{R}$ such that $\forall x, y \subseteq E$:

$$\delta(x, y) = \frac{|x \cup y - x \cap y|}{|x \cup y|}$$

Instance identification techniques

- If common set of instances is not available
- We can try to match instances
- One way is to use keys (in ideal case they are not internal to the database but external - i.e. isbn by books, or ean by some goods)
- Other approaches use the instance data to compare property values
- Usually string-based and structure-based techniques are used

Disjoint extension comparison

- Approximate techniques for comparing class extensions
- **Statistical approach:** Compute some statistics about the property values found in instance (maximum, minimum, mean, existence of null values etc.) - this allows the characterising of the domains of class properties. Or utilise data patterns and distribution instead of data values and domains.
- **Similarity-based extension comparison:** Compare two classes on the basis of their instances - compare two sets of instances and compute the dissimilarity of the two sets
- **Matching based comparison:** Also two sets of instances are compared, but only corresponding elements are compared with each other

Techniques based on external ontologies

- Semantic based techniques are deductive methods $\Rightarrow$ they need a preprocessing phase which provides 'anchors' (entities which are declared, for example, to be equivalent)
- Approach: Using intermediate formal ontologies, which can define the common context or background knowledge
- **Anchoring:** Matching ontologies o' and o'' to the background ontology o
- **Deriving relations:** Matching of ontologies o' and o'' by using the correspondence discovered during the anchoring step

Deductive techniques

- The merging of two ontologies and the search for correspondence A such that $o, o' \models A$
- These techniques can also be used for testing the satisfiability of alignments
- **Propositional techniques:**
  1. Build the theory or domain knowledge (*Axioms*) for the given two ontologies as a conjunction of the available axioms
  2. Build a matching formula for each pair of classes c and c'
  3. Check for validity of the formula (that it is true for all the truth assignments of all the propositional variables occurring in it)
- **Description logic techniques:** The relations (e.g. $=$, $\sqsubseteq$, $\sqsupseteq$, $\perp$) can be expressed with respect to subsumption

Propositional techniques - Example

Starting situation

First ontology: classes Europe and images
Second ontology: classes Europe and pictures
We already know that *pictures* $\leftrightarrow$ *images* and
Europe $\leftrightarrow$ *Europe*

1. Translate the relations into Axioms:
   $(pictures \leftrightarrow images) \wedge (Europe \leftrightarrow Europe)$
2. There are defined two concepts: c as $Europe \sqcap images$ and c' as $pictures \sqcap Europe$. **Is c equivalent to c'?**
   $(images \leftrightarrow pictures) \wedge (Europe \leftrightarrow Europe)) \rightarrow ((Europe \wedge images) \leftrightarrow (Europe \wedge pictures))$
3. Is the formula satisfiable? Is the negation unsatisfiable?

Summary

- We introduced the **string comparison techniques** which are useful if people use very similar strings to denote the same concepts (many synonyms with different structures yield a low similarity).
- Problem of synonyms can be partly solved with help of **linguistic methods**. Unfortunately, they also recognise that the same term may denote several concepts at once.
- One way to choose among these representations is to take into account the **structure of ontology** entities.
- **Extension information**, contrary, is independent from conceptual part of ontology. Extension is supposed to be less prone to variability and can be used to accurate match the classes.
- Finally the **semantic techniques** can be used to ensure the completeness and the consistency of the alignment.
- All the discussed techniques can not be used in isolation, but each of them can take advantage of the results provided by others.

Questions?

???

Thanks

Thanks for your attention

References

- Jérôme Euzenat and Pavel Shvaiko: Ontology Matching, Springer-Verlag, 2007.
- Ian Niles, Adam Pease: Towards a Standard Upper Ontology.
- AnHai Doan, Jayant Madhavan, Pedro Domingos, and Alon Halevy: Ontology Matching: A Machine Learning Approach.
- Jérôme Euzenat and Pavel Shvaiko: Tutorial on Schema and Ontology Matching, 2005.
- Ontology Matching Website, http://ontologymatching.org/, 05 2008.
- SimMetrics java library web page, http://www.dcs.shef.ac.uk/ sam/stringmetrics.html, 06 2008.
- The suggested upper method ontology (SUMO), http://www.ontologyportal.org/, 05 2008.
- Laboratory for Applied Ontology - DOLCE, http://www.loa-cnr.it/DOLCE.html, 05 2008.
- WordNet - Princeton University Cognitive Science Laboratory, http://wordnet.princeton.edu/, 05 2008.