Ontologies and Matching Techniques for Knowledge Sharing and Evolution in P2P Systems

S. Castano, A. Ferrara, S. Montanelli. G.Racca
DICO – Università degli Studi di Milano
[castano, ferrara, montanelli,racca]@dico.unimi.it

Bologna, 11 dicembre 2003

Talk by S. Castano

Overview

- Knowledge sharing in distributed systems
- Reference ontology model
- Metadata repository
- Ontology matching techniques
- Application to the HELIOS peer-based system
- Ongoing research

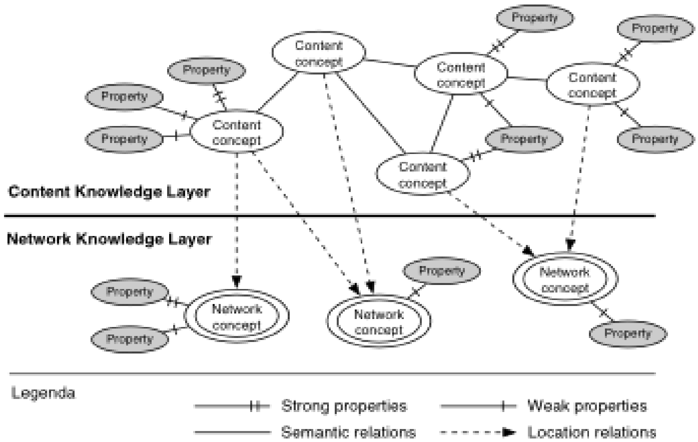
Knowledge sharing in distributed systems

- Resources to be shared are provided by many different nodes, possibly spanned across organizations
- A central problem is dynamic resource discovery, that is, the capability of dynamically finding the distributed resources that best match a given target request
- Ontologies are generally recognized as an essential tool for allowing communication and knowledge sharing among distributed users and applications (e.g., Semantic Web efforts)
- In distributed systems, knowledge of interest is provided by many different nodes with autonomous and possibly heterogeneous ontologies
- To enforce dynamic resource discovery, appropriate ontology matching techniques are required

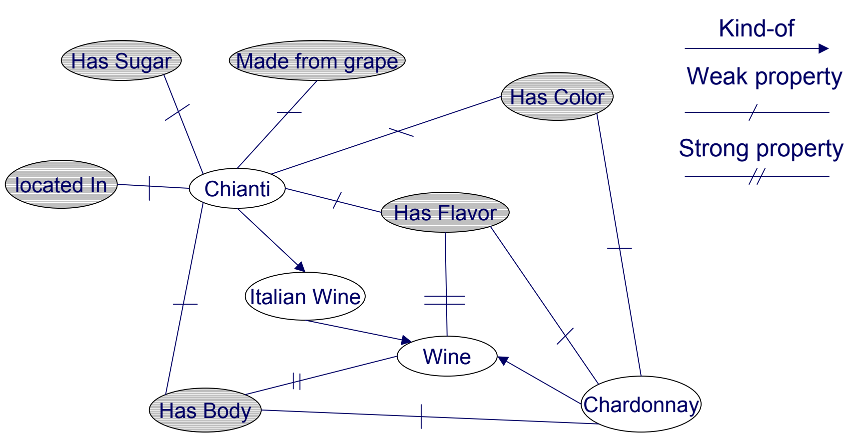
Ontologies and resource description

- *An ontology is an explicit specification of a conceptualization [Gruber, 1993]*
- *An ontology is a shared understanding of some domain of interest [Uschold&Gruninger, 1996]*
- Using ontology allows one to exploit Semantic Web formalisms for knowledge representation, to provide a semantically rich description of the information of interest in terms of concepts, properties, semantic relations, constraints.
- Knowledge of interest
  - About the resources a node brings to the network
  - About the nodes of the system for the effective routing of target requests depending on the goal to be pursued (e.g., knowledge on my semantic neighbors in a P2P network to support dynamic thematic communities/semantic routing protocols)

Reference ontology model



Example



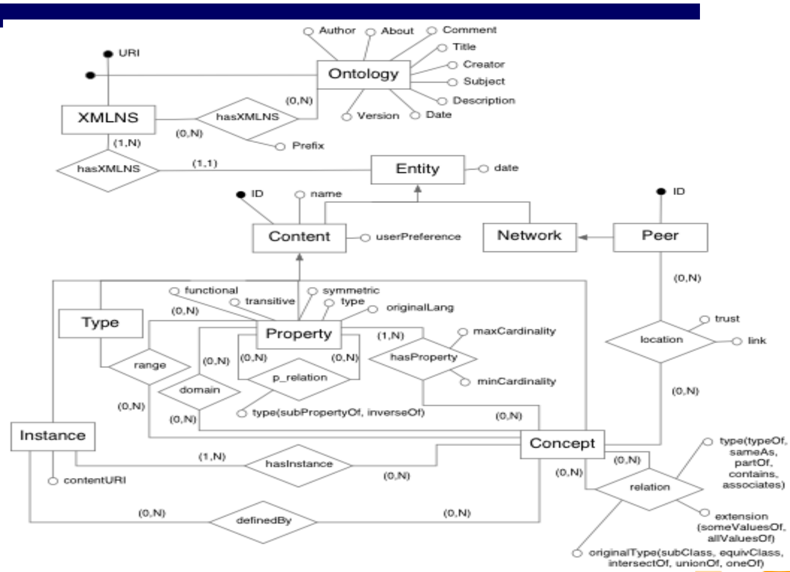
Portion of Wine ontology (http://www.w3.org/TR/2003/CR-owl-guide-20030818/wine.rdf)

Ontology manager

- **Ontology definition**: construction of the node ontology
  - Wrapping for main Semantic Web standards for ontology languages (e.g., OWL, RDF(S))
  - Metadata repository for storing the ontology contents
  - Capability of building a node ontology starting from several content sources of the node (e.g., use of ARTEMIS tool environment)

- **Ontology evolution**: to enrich the knowledge in a node ontology by adding new concepts or by extending existing concepts with new knowledge acquired by other peers

Metadata repository schema



Ontology matching techniques

- The aim is to find concept(s) semantically related to one or more target concept(s), using ontology descriptions
- **Dynamic matching**
  - at different levels of depth with different degrees of flexibility and accuracy of results
  - taking into account different levels of richness in resource descriptions
  - considering various metadata elements separately or in combination

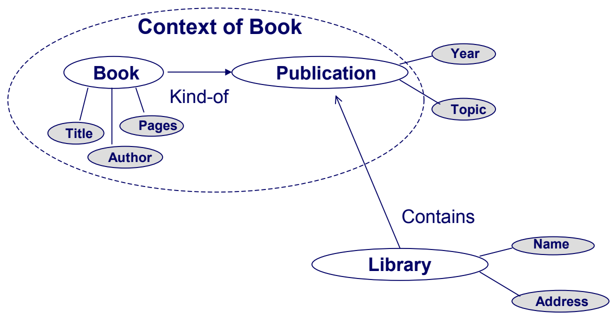
H-MATCH algorithm which considers both the linguistic features and contextual features of concept descriptions

H-MATCH: linguistic interpretation

- Linguistic features are constituted by the semantic content of terms used as names of concepts and properties
- A thesaurus of terminological relationships is built by exploiting WordNet
- A subset of the relationships provided by WordNet is considered:
  - Synonyms, SYN
  - Hypernyms, BT
  - Hyponyms, NT
  - Meronyms, Holonyms, RT

H-MATCH: context interpretation

- H-MATCH explicitly considers the context of each concept as the set of its properties and its adjacents (i.e., concepts having a semantic relation with the considered concept)



H-MATCH: semantic affinity evaluation

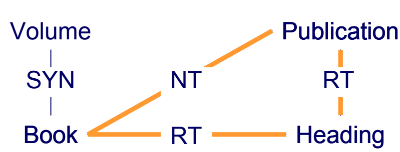
- The semantic affinity of two concepts is a linear combination of a linguistc affinity and a contextual affinity
- Terminological relationships in the thesaurus and the semantic relations in the concept contexts are weighted
  - The weights associated with terminological relationships have been tested on several real integration cases in the ARTEMIS tool environment
- The weights associated with semantic relations express a measure of the strength of concept connections
  - We are tuning these weights on real ontology matching cases in the HELIOS framework

H-MATCH: linguistic affinity

- The aim is to evaluate a measure of affinity between concepts based on their names
- The linguistic affinity of two concepts is equal to the highest-strength path of terminological relationships between their names in the thesaurus
- The linguistic affinity of two concepts is zero is no path exists between their names in the thesaurus

H-MATCH: linguistic affinity

Example of linguistic affinity evaluation:



LA(Book,Publication) ??

First Path = [NT] = 0.8

Second Path = ~~[RT,RT] = 0.5~~ x 0.5 = 0.25

LA(Book,Publication) = 0.8

H-MATCH: contextual affinity

- The aim is to evaluate a measure of affinity between concepts based on their contexts
- The linguistic affinity of properties and adjacents is considered as well as the degree of closeness of the semantic relations in the concept contexts
- The relation closeness is a value X in the range [0,1] and is evaluated as:

  $X = 1 - |\sigma_r - \sigma_{r'}|$, where σ_r and $\sigma_{r'}$ are the weights associated with the semantic relations r and r', respectively
- The higher the difference between the weights associated with the relations, the lower the closeness between them

H-MATCH: models for dynamic matching

- **Shallow matching**

  Considers only the linguistic information provided by the concept names and by the reference thesaurus.
- **Intermediate matching**

  Considers concept names and also properties of the concept context.
- **Deep matching**

  Considers concept names and the whole context of concepts (i.e., properties and adjacents).

The algorithm (1)

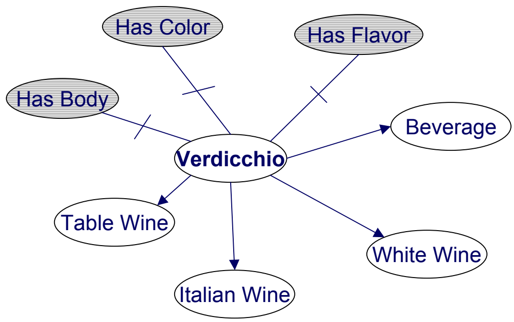
algorithm H-MATCH($c, c', model =$ "deep"$, W_{LA} = 0.5$)
input the concepts c and c', the matching model $\in$ [shallow; intermediate; deep], and the weight $W_{LA} \in$ [0,1]
output the semantic affinity value between c and c'
begin algorithm
  def t, t' as the names of c and c', respectively;
  def $CV(c) = [], CV(c') = []$ as the context vectors for c and c', respectively;
  def *context_item* = [] as a pair of the form (f, r), where f is a name associated with a property or a concept, and $r \in$ {property; same-as; kind-of; part-of; contains; associates};
  def $x = 0, y = 0, semantic\_affinity = 0$;
  $x = LA(t, t')$;
  switch *model*
    case "shallow" :
      $W_{LA} = 1$;
    case "intermediate" :
      foreach property $p(c) \in Ctx(c)$
        context_item = [$p(c)$, "property"];
        append *context_item* to $CV(c)$;
      foreach property $p(c') \in Ctx(c')$
        context_item = [$p(c')$, "property"];
        append *context_item* to $CV(c')$;

The algorithm (2)

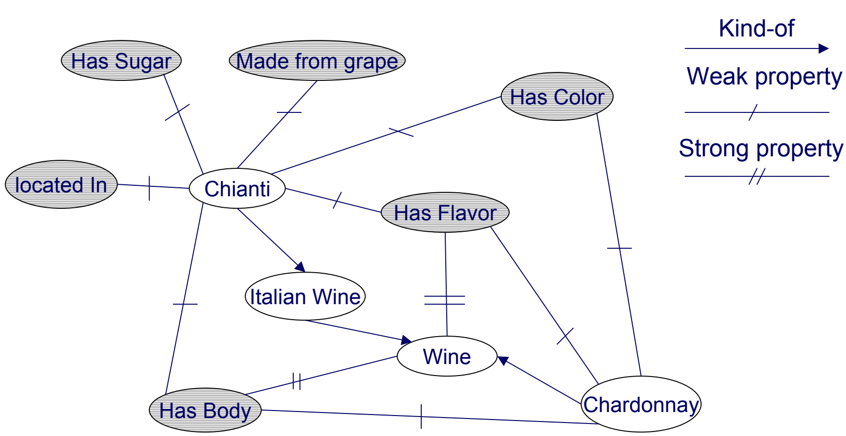
    case "deep" :
      foreach property $p(c) \in Ctx(c)$
        context_item = [$p(c)$, "property"];
        append *context_item* to $CV(c)$;
      foreach concept $c_i \in Ctx(c)$
        /* $sr(c, c_i)$ is the semantic relation between c and c_i */
        context_item = [$c_i, sr(c, c_i)$];
        append *context_item* to $CV(c)$;
      foreach property $p(c') \in Ctx(c')$
        context_item = [$p(c')$, "property"];
        append *context_item* to $CV(c')$;
      foreach concept $c_j \in Ctx(c')$
        /* $sr(c', c_j)$ is the semantic relation between c' and c_j */
        context_item = [$c_j, sr(c', c_j)$];
        append *context_item* to $CV(c')$;
  $y = CA(CV(c), CV(c'))$;
  $semantic\_affinity = W_{LA} \cdot x + (1 - W_{LA}) \cdot y$;
  return *semantic_affinity*;
end algorithm

Example of matching

Target



Example



The Wine ontology contains 200 concepts

Matching results

Semantic Affinity	*Shallow*	*Intermediate*	*Deep*
SA(Wine, Verdicchio)	**0.64**	**0.447829**	**0.474677**
SA(Italian Wine, Verdicchio)	0.512	0.256	0.488
SA(Chardonnay,Verdicchio)	**0.64**	**0.82**	**0.742286**
SA(Chianti, Verdicchio)	0.4096	0.484267	0.6112
SA(Red Wine, Verdicchio)	0.512	0.256	0.488
SA(White Wine, Verdicchio)	**0.8**	**0.4**	**0.632**

Related work

Matching techniques have been investigated in both data integration and ontology literature:

- **CUPID** (schema matching), Bernstein et al., VLDB 2001
- **Edamok** (SAT-based), Serafini et al., CONTEXT 2003
- **Chatty Web** (mapping based agreement), Aberer et al., WWW 2003
- **GLUE** (machine learning), Halevy et al., WWW 2002

The matching problem is addressed also in peer-based distributed systems, such as:

- **Edutella**, Nejdl et al., WWW 2003
- **KAON**, Motik et al., ODBASE 2002

Knowledge sharing in HELIOS

- HELIOS (Helios Evolving Interaction-based Ontology knowledge Sharing) is a framework for knowledge sharing and evolution in P2P networks
- The knowledge sharing and evolution processes in HELIOS are based on:
  - **peer ontologies**, describing the knowledge each peer brings to the network and/or has acquired from the network, in terms of metadata
  - **interactions among peers**, allowing information search and knowledge acquisition/extension according to pre-defined query models
  - **ontology matching** techniques

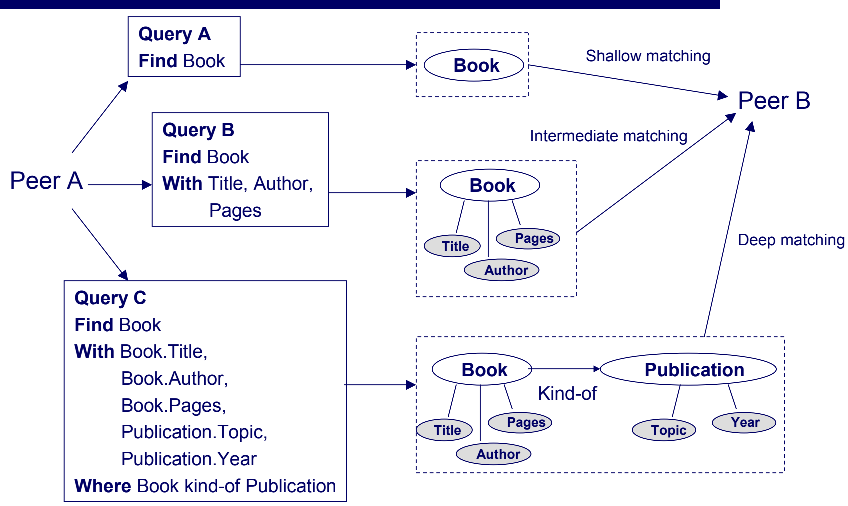
HELIOS

- Implements a peer ontology with a metadata repository according to the reference ontology model
- Uses H-MATCH for ontology matching in query resolution
- Allows peers to form communities of interest and share their knowledge in spite of their network distance (*notion of semantic neighbourhood*), by exploiting pre-defined query models and a semantic routing protocol (under development in GL3, joint work with the NPTLab at UNIMI).

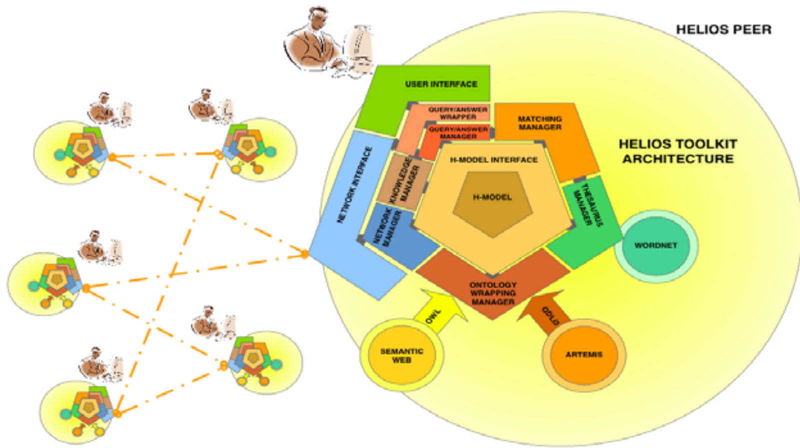
HELIOS query models

- **Probe model:** used to acquire knowledge from the network. The answer is a set of metadata.
- **Search model:** used to find contents related to one or more concepts of interest. The answer is a set of data.
- **Probe/Search model:** used to extend knowledge and find contents. The answer is a set of data and metadata.
- Query resolution
  - Extraction of an ontological description of target request in the query
  - Matching the target against the peer ontology in order to find a set of semantically related concepts
  - Answer construction

Example



HELIOS architecture



Joining the HELIOS network

- Initialization
  - The new peer receives the HELIOS toolkit for query composition and query resolution, and for peer ontology definition. At the end of this initialization phase, the peer is able to interact with the other HELIOS peers
  - The new peer defines the peer ontology describing its data
- Notification
  - **Import approach:** the new peer submits to the network a set of probe queries concerning the concepts in its ontology
  - **Export approach:** the new peer sends to other Helios peers the description of the concepts in its ontology
  - **Passive approach:** the new peer does not perform its notification. The peer knowledge will be discovered by the probe queries sent by the other peers

Ongoing research

- Metadata repository implementation with RDB
- Extensive testing of H-MATCH on several real ontologies acquired from the web, to increase performance and scalability
- Ontology knowledge evolution
  - Studying techniques to support ontology evolution at different levels of severity (e.g., importation vs. mapping vs. integration of new concepts)
- Development of the semantic routing protocol and support for dynamic thematic communities management (jointly with NPTLab)