3
Catalogue matching

A *catalogue* is a simple database that holds information about a set of objects, typically classified using terms taken from a given thesaurus. Catalogues are fairly common and can be found, for example, in e-commerce and GIS applications, such as on-line stores and gazetteers. The schema of a catalogue has a single class with a list of properties. Matching a pair of catalogues raises three problems: (1) to match their conceptual schemas; (2) to find a relationship between the terms of their thesauri; (3) to define a way of identifying when two objects from different catalogues represent the same real-world object. Note that the last two problems are usually not considered in database schema matching.

The major contributions of this chapter are three-fold. First, we introduce a matching approach, based on the notion of similarity, which applies to pairs of thesauri and to pairs of lists of properties. Second, we describe matchings based on cooccurrence of information and introduce variations that explore certain heuristics. Third, we discuss experimental results that evaluate the precision of the matchings introduced and that measure the influence of the heuristics.

3.1.
Catalogues, catalogue queries and catalogue matching

A *thesaurus* is defined as "a structured and defined list of terms which standardizes words used for indexing" (UNESCO 1995) or, equivalently, "*the vocabulary of a controlled indexing language, formally organized so that a priori relationships between concepts (for example as "broader" and "narrower") are made explicit"* (ISO2788 1986). A thesaurus usually provides: a *preferred term,* defined as the term used consistently to represent a given concept; a *non-preferred term,* defined as the synonym or quasi-synonym of a preferred term; relationships between the terms, such as narrower term (NT), indicating that a term – the narrower term – refers to a concept which has a more specific meaning than another term – the broader term (BT).

A *catalogue* is a simple database that holds information about a set of *object instances*, or simply *instances*.

A *catalogue schema* consists of a *type thesaurus T* and an OWL schema, denoted in abbreviated form as $C[A_1\ U_1, ..., A_m\ U_m]$, where

- C is the name of the single *class* of the schema
- A_1 is the *id*, which is an inverse functional property (a key) that uniquely identifies the instances stored in the catalogue
- A_2 is the *type*, which is a functional property whose range is the set of terms in T (for simplicity, we assume that the type is therefore unique)
- $A_3, ..., A_m$ is a possibly empty list of distinct datatype properties
- U_i is the range of property A_i, for each $i \in [1,m]$, which for simplicity we assume to be a subset of the *universe U* (the range of the *type* property is the set of terms of the thesaurus T)

We also say that *T is the thesaurus of the schema* $C[A_1\ U_1, ..., A_m\ U_m]$.

The *universe* of C is the set U_C of all possible RDF triples that conform to the schema. An *extension* of C is a subset of U_C.

A *conjunctive restriction query* over C is a conjunction of *restriction predicates*, defined over the properties of C. A *restriction predicate* over C is an expression of the form $A_i=v$, where A_i is a property of C and v denotes a value in U_i. Informal examples of conjunctive restriction queries over a catalogue of household appliances would be “select all 17 inch flat panel TVs” and “select all 220V food processors”.

Catalogues usually provide a simple user interface that supports conjunctive restriction queries and that organizes the query results as lists of instances, which the user may browse and select the instances that catch his attention. Catalogues recently started to expose such functionality through Web services.

Let $C[A_1\ U_1, ..., A_m\ U_m]$ and $D[B_1\ V_1, ..., B_n\ V_n]$ be two catalogue schemas with thesauri T and W, respectively.

A *property matching* between C and D is a partial, many-to-many relation $\mu_P \subseteq \{A_3,...,A_m\} \times \{B_3,...,B_n\}$. We allow μ_P to be partial since some property of C may not match any property of D, and vice-versa, and we let μ_P to be many-to-many to account for properties from C that match several properties of D, and vice-versa. We say that μ_P is *unambiguous* iff μ_P is one-to-one. Likewise, a

thesaurus matching between *C* and *D* is a partial, many-to-many relation μ_T between terms of *T* and terms of *W*. An *instance matching* between *C* and *D* is a partial, many-to-many relation μ_I between subjects of triples in U_C and subjects of triples in U_D.

We say that an instance *I* in U_C *matches* an instance *J* in U_D iff $(I,J) \in \mu_I$, and likewise for properties and thesauri terms.

A *matching* between *C* and *D* is a triple (μ_P,μ_T,μ_I) such that μ_P is a property matching, μ_T is a thesaurus matching and μ_I is an instance matching between *C* and *D*.

3.2. An informal example of catalogue matching

The area of Geographic Information Systems (GIS) provides interesting applications of catalogues, which we explore in this section to construct an example of thesauri matching. We close the section with two brief comments on property and instance matching.

A gazetteer is "a geographical dictionary (as at the back of an atlas) containing a list of geographic names, together with their geographic locations and other descriptive information" (WordNet 2005). For our purposes and omitting details, we consider that a gazetteer is a catalogue of geographic locations, where each instance has as properties:

- a unique *ID*;
- a unique *type*, whose value is a term taken from a *type thesaurus*;
- a *name*, which takes a character string as value;
- optionally, a *position*, which approximates the position of the geographic location on the Earth's surface.

Consistently with Section 3.1, we assume that the type is unique. We note that geographic locations are often called *geographic features*, or simply *features* (Percivall 2003). Hence, a gazetteer thesaurus is often referred to as a *feature type thesaurus*.

Almost all gazetteers support conjunctive restriction queries using type and name restrictions, such as "select all populated places called 'Rio de Janeiro' ", where 'populated place' is a term of the feature type thesaurus. Some gazetteers

also allow conjunctive restriction queries that include spatial restrictions, such as "find all populated places within 10 miles of point *P*", where *P* is defined in an appropriate coordinate (geo)reference system.

Specifically, in our example, we will use two gazetteers that are available over the Web, the GEOnet Names Server and the Alexandria Digital Library Gazetteer. The GEOnet Names Server (GNIS 2005) provides access to the National Geospatial-Intelligence Agency (NGA) and the U.S. database of foreign geographic names, containing about 4 million features with 5.5 million names. The Alexandria Digital Library (ADL) Project (Hill et al. 1999, Janée and Hill 2004, ADL 1999) is a research program to model, prototype and evaluate digital library architectures, gazetteer applications, educational applications, and software components. The ADL gazetteer has approximately 5.9 million geographic names, classified according to the ADL Feature Type Thesaurus (FTT).

Figure 2 (a) shows a fragment of the ADL Feature Type Thesaurus and Figure 2 (b) contains the equivalent fragment of the GEOnet Names Server classification scheme, which strictly speaking is not a thesaurus, but just a list of terms without any thesauri relationships. Note that a simple thesauri matching strategy, based on syntactical proximity, would be of little help to match the ADL Feature Type Thesaurus and the GEOnet Names Server classification scheme since the latter uses codes as thesauri terms.

In what follows, we will refer to the ADL gazetteer and the GEOnet Names Server, respectively, as ADL and GNS, and to their thesauri as ADL FTT (for ADL Feature Type Thesaurus) and GNS CS (for GEOnet Names Server classification scheme). We will consider only countries and cities in the examples that follow. For simplicity, we assume that the name (in English) uniquely

Administrative Area
….Populated Places
….Cities
……..Capitals
….Political Areas
……..Countries

(a) ADL FTT fragment.

Code	Description Text
PCLI	"Independent political entity"
AREA	"A tract of land without homogeneous character or boundaries"
PPL	"Populated place"
PPLA	"Seat of a first-order administrative division"
PPLC	"Capital of a political entity"
PCLI	"Independent political entity"

(b) GEOnet Classification Scheme fragment.

Figure 2. Fragments of the ADL and GEOnet thesauri.

identifies a country in both catalogues; similarly, the city name, together with the name of the upper level administrative division, uniquely identifies a city in both catalogues.

We will illustrate how to gradually construct a matching between these thesauri by post processing the answers to queries submitted to both gazetteers. The matching may help construct a mediator to access both gazetteers or to consolidate them in a single gazetteer (as in a data warehouse application) by remapping their thesauri.

Table 1 shows sample terms collected from queries that searched the two gazetteers for the countries and cities listed in the first column. For example, if we query ADL to obtain information about “Brazil”, the answer will indicate that ADL classifies “Brazil” as “Countries”; if we then access GNS for “Brazil”, the answer shows that GNS classifies “Brazil” as “PCLI”. Therefore, we collected the first evidence that these two terms map to each other.

In fact, all 5 entries in Table 1 that ADL classifies as “Countries”, GNS classifies them as “PCLI”. Hence, we have better evidence that these two terms map to each other since they refer to the same five countries. If we consider that the meaning of a thesaurus term is the set of objects it classifies, then “Countries”

Table 1. Results of querying countries and cities in *ADL* and *GNS*.

Entry name	ADL	GNS
Brazil	Countries	PCLI
Canada	Countries	PCLI
Germany	Countries	PCLI
Italy	Countries	PCLI
Belgium	Countries	PCLI
Scotland – UK	AdministrativeArea	AREA
Wales – UK	AdministrativeArea	AREA
Rio Grande – Brazil	Populated Places	PPL
Smithers – Canada	Populated Places	PPL
Rio de Janeiro – Brazil	Populated Places	PPLA
São Paulo – Brazil	Populated Places	PPL
Cardiff – Wales	Populated Places	PPLA
Asmara – Eritrea	Capitals	PPLC
Rome – Italy	Capitals	PPLC
Brussels – Belgium	Capitals	PPLC

and “PCLI” have the same meaning in this small sample. Moreover, we have not detected any conflicting classifications. The question then is how many mismatches we should allow, that is, how similar the sets of geographic locations that two thesauri terms denote must be to consider that the two terms match.

To better explain this last remark, observe now the entries in Table 1 that ADL classifies as “Populated Places”. Note that GNS classifies three of them as “PPL” and two as “PPLA”. There are two approaches to address such situation. We may decide to match “Populated Places” with both “PPL” and “PPLA”. That is, we may decide to allow 1-to-many matchings. Alternatively, we interpret the evidence collected thus far as an indication that “Populated Places” matches “PPL” better (three entries in Table 1) than “PPLA” (two entries in Table 1).

The key questions therefore are what “to collect enough evidence” means, and what to do when the mapping between terms is not one-to-one. This is addressed in Section 3.4 by introducing similarity functions that estimate how close the sets of instances that two terms denote are.

We may adopt the same strategy to match the properties of ADL and GNS, but we would now collect information about the property values from the query answers. In other words, we argue that the problem of matching thesauri terms and the problem of matching properties may be both reduced to measuring set similarity: (1) similarity between the sets of instances the terms classify, in the former case; and (2) similarity between the sets of property values, in the latter case. Section 3.5 contains results about the effectiveness of this strategy.

We close with a brief comment on the problem of instance matching. In the geographic information systems domain, we have various geo-referencing schemes that associate each geographic location with a description of its position on the Earth’s surface. This position acts as a universal identifier for the object, or at least an approximation thereof. In this case, we may propose that two instances match if their positions and their names are similar. This strategy works well for the geographic domain, but it depends on detecting – and matching – which properties describe the geographic position and the name of the instances in both gazetteers. In general, one will use some form of comparing property values to induce instance matchings. However, the user will typically have to interfere to inform which properties to use (such as the position and the name) and how to compare them.

3.3. Matching heuristics

As illustrated in Section 3.2, extensional matching techniques use duplicated values to formulate hypothesis about the matching. In this section, we expand this observation, discuss five practical heuristics to compute μ_A and introduce the notion of similarity-induced catalogue matching.

Let $C[A_1\ U_1, ..., A_m\ U_m]$, with thesaurus T, and $D[B_1\ V_1, ..., B_n\ V_n]$, with thesaurus W, be the *source* and the *target catalogue schemas*.

Let $\boldsymbol{C}$ and $\boldsymbol{D}$ be extensions of source and target catalogues.

The *observed domain heuristic* considers that the larger the cardinality of the intersection of the set of observed values in $\boldsymbol{C}$ of a property A_i in C with the set of observed values in $\boldsymbol{D}$ of a property B_j in D, the higher the expectation that A_i and B_j represent the same concept.

To exemplify this heuristic, consider the fragments of a source and target book catalogues, expressed as RDF triples and depicted in Figure 3. The source catalogue has a single property *title* and the target catalogue has properties *title* and *author*. It is more likely that `source:title` matches `target:title` than `target:author` because they have 4 values in common (“King Lear”, “Romeo and Juliet”, “Hamlet”, “Macbeth”, “Dom Casmurro”) in a total of 6

C (source)		
S	P	O
10	`title`	“King Lear”
20	`title`	“Romeo and Juliet”
30	`title`	“Hamlet”
40	`title`	“Othello”
50	`title`	“Dom Casmurro”

D (target)		
S	P	O
100	`title`	“King Lear”
200	`title`	“Romeo and Juliet”
300	`title`	“Hamlet”
400	`title`	“Macbeth”
500	`title`	“Dom Casmurro”
600	`title`	“Quincas Borba”
500	`author`	“Machado de Assis”
600	`author`	“Machado de Assis”
100	`author`	“William Shakespeare”
200	`author`	“William Shakespeare”
300	`author`	“William Shakespeare”
400	`author`	“William Shakespeare”

Figure 3. RDF triples of fragments of book catalogues.

different values for `source:title` and `target:title`, i.e., 67% of *commonalities* (number of common values divided by the total number of values of both properties), while `target:author` does not have anything in common with `source:title`.

The *commonality*, as presented above, is in fact a measure of the degree of similarity between the sets of observed values. We introduced this concept to avoid discussions on more elaborate similarity measures. We do not use the commonality measure in the experiments we conducted to evaluate the matching technique. So that, we define that two concepts are *semantically equivalent* iff the similarity of the sets of their observed values is above a given threshold.

Let A_i be a property of C. The *observed domain representation of* A_i in $\boldsymbol{C}$ is the set $\boldsymbol{o}[\boldsymbol{C},A_i]$ such that $v \in \boldsymbol{o}[\boldsymbol{C},A_i]$ iff there is a triple of the form (I,A_i,v) in $\boldsymbol{C}$ (that is, there is an instance I in $\boldsymbol{C}$ such that the value of A_i for I is v). The observed domain representation of a property of D is likewise defined.

We refer to $\boldsymbol{o}[\boldsymbol{C},A_i]$ and $\boldsymbol{o}[\boldsymbol{D},B_j]$ as the *characteristic sets* of A_i and B_j, respectively. As we will discuss in what follows, properties may have other characteristic sets. Hence, more elaborated matching models may use, in conjunction, a collection of such characteristic sets.

Let $\boldsymbol{U}$ denote the universe of property values. Consider a (generic) similarity function $\sigma: 2^{\boldsymbol{U}} \times 2^{\boldsymbol{U}} \rightarrow \mathbb{R}^+$ over $2^{\boldsymbol{U}}$ and a positive Real number, τ, the *similarity threshold* for sets of values in $\boldsymbol{U}$. We define the *property matching between C and D induced by* σ *and* τ as the partial, many-to-many relation

$$\mu_P \subseteq \{A_1,...,A_m\} \times \{B_1,...,B_n\} \text{ such that } (A_i,B_j) \in \mu_A \text{ iff } \sigma(\boldsymbol{o}[\boldsymbol{C},A_i],\boldsymbol{o}[\boldsymbol{D},B_j]) \geq \tau.$$

Let t be a term of the thesaurus T. The *representation* of t in $\boldsymbol{C}$ is the set $\boldsymbol{i}[\boldsymbol{C},t]$ such that $I \in \boldsymbol{i}[\boldsymbol{C},t]$ iff there is a triple of the form $(I,type,t)$ in $\boldsymbol{C}$ (that is, there is an instance I in $\boldsymbol{C}$ such that the type of I is t). The representation of a term of W is likewise defined..

Let $\boldsymbol{I}$ be the set of all instances of $\boldsymbol{C}$ and $\boldsymbol{D}$. Consider a (generic) similarity function $\sigma: 2^{\boldsymbol{I}} \times 2^{\boldsymbol{I}} \rightarrow \mathbb{R}^+$ over $2^{\boldsymbol{I}}$ and a positive Real number, τ, the *similarity threshold* for sets of values in $\boldsymbol{I}$. We define the *thesauri matching between T and W induced by* σ *and* τ as the partial, many-to-many relation

C (source)		
S	P	O
10	`title`	“King Lear”
20	`title`	“Romeo and Juliet”
30	`title`	“Hamlet”
40	`title`	“Othello”
50	`title`	“Dom Casmurro”

D (target)		
S	P	O
100	`title`	“King Lear **(new)**”
200	`title`	“Romeo and Juliet **(Arkangel Complete Shakespeare)**”
300	`title`	“Hamlet”
400	`title`	“Macbeth”
500	`title`	“Dom Casmurro”
600	`title`	“Quincas Borba”
500	`author`	“Machado de Assis”
600	`author`	“Machado de Assis”
100	`author`	“William Shakespeare”
200	`author`	“William Shakespeare”
300	`author`	“William Shakespeare”
400	`author`	“William Shakespeare”

Figure 5. RDF triples of fragments of book catalogues.

$$\mu_T \subseteq \{t_1,...,t_m\} \times \{u_1,...,u_n\} \text{ such that } (t_i,u_j) \in \mu_T \text{ iff } \sigma(\boldsymbol{i}[\boldsymbol{C},t_i],\boldsymbol{i}[\boldsymbol{D},u_j]) \geq \tau.$$

Let us now return to property matchings. Consider, in Figure 5, the same catalogue fragments of Figure 3, except that the first two rows of the target extension have been changed. We may now be lead to the wrong conclusion that properties `source:title` and `target:title` are not semantically equivalent, depending on a given threshold, because their commonalities decreased to just 22% (2 common values in the total of 9 different values). However, if we consider the sets of observed tokens of these properties, the similarity between them becomes again evident (see Figure 4), since it increases to 50% (7 common tokens in the total of 14 tokens of both properties).

`source:title` ∈ {King, Lear, Romeo, Juliet, Hamlet, Othello, Dom, Casmurro}

`target:title` ∈ {King, Lear, Romeo, Juliet, Arkangel, Complete, Shakespeare, Hamlet, Macbeth, Dom, Casmurro, Quincas, Borba}

`source:title, target:title` → 50% of commonality

Figure 4. Observed tokens of the title property in the source and target databases of Figure 5.

The set of tokens of a string *s* is obtained as follows. First, tokens are extracted by splitting *s* into each non-word or non-numeric characters to obtain a set of substrings. Then, this set is reduced by eliminating substrings which are stop-words. Finally, the remaining strings are lemmatized (Manning and Schütze 2002).

Let A_i be a character string property of *C*. The *observed tokens representation of* A_i in $\boldsymbol{C}$ is the set $\boldsymbol{ot}[\boldsymbol{C},A_i]$ such that $t \in \boldsymbol{ot}[\boldsymbol{C},A_i]$ iff there is a triple of the form (I,A_i,v) in $\boldsymbol{C}$ such that *t* is a token of *v* (that is, there is an instance *I* in $\boldsymbol{C}$ such that the value of A_i for *I* is *v* and *t* is a token of *v*). The observed tokens representation of a character string property of *D* is likewise defined.

Let $\boldsymbol{T}$ denote the universe of tokens. Consider a (generic) similarity function $\sigma: 2^{\boldsymbol{T}} \times 2^{\boldsymbol{T}} \rightarrow \mathbb{R}^+$ over $2^{\boldsymbol{T}}$ and a positive Real number, τ, the *similarity threshold* for sets of values in $\boldsymbol{T}$. We define the *property matching for character string properties between C and D induced by* σ *and* τ as the partial, many-to-many relation

$$\mu_P \subseteq \{A_1,...,A_m\} \times \{B_1,...,B_n\} \text{ such that } (A_i,B_j) \in \mu_A \text{ iff } \sigma(\boldsymbol{ot}[\boldsymbol{C},A_i],\boldsymbol{ot}[\boldsymbol{D},B_j]) \geq \tau.$$

We refer to the form of measuring the similarity of pairs U_i and V_j of properties of type *string* by computing $\sigma(\boldsymbol{ot}[\boldsymbol{C},A_i],\boldsymbol{ot}[\boldsymbol{D},B_j])$ as the *string domain heuristic*.

From this point on in this thesis, we unify the observed domain and observed token heuristics by redefining the observed domain $\boldsymbol{o}[\boldsymbol{C},A_i]$ for string properties as $\boldsymbol{ot}[\boldsymbol{C},A_i]$.

The third heuristic, called *instance matching heuristic* is not so obvious. To explain it, consider the two sets of RDF triples about books in Figure 6. The schemas have only four properties each: *isbn*, *title*, *edition* and *rating*. The *rating* property indicates the popularity of the book. The three triples at the end of each set of triples means that the instances of the database referred as the subjects of the triples are equivalent to the instances referred as the objects in the other database.

Using the technique of comparing observed values or tokens, we have the induced matching shown in Figure 7. The last two underscored lines are *false positive* matchings, because the observed values for *edition* and *rating* are equal.

C (source)		
S	P	O
10	isbn	1434610985
10	edition	1
10	rating	2
10	title	“King Lear”
20	isbn	0140867724
20	edition	2
20	rating	3
20	title	“Romeo and Juliet”
30	isbn	0140714634
30	edition	3
30	rating	4
30	title	“Othello”
40	isbn	0195103092
40	edition	4
40	rating	1
40	title	“Dom Casmurro”
10	owl:sameAs	100
20	owl:sameAs	200
40	owl:sameAs	400

D (target)		
S	P	O
100	isbn	1434610985
100	edition	1
100	rating	2
100	title	“King Lear”
200	isbn	0140867724
200	edition	2
200	rating	3
200	title	“Romeo and Juliet”
300	isbn	0140714782
300	edition	3
300	rating	4
300	title	“Macbeth”
400	isbn	0195103092
400	edition	4
400	rating	1
400	title	“Dom Casmurro”
100	owl:sameAs	10
200	owl:sameAs	20
400	owl:sameAs	40

Figure 6. RDF triples of fragments of book catalogues with instance matching.

Let us now change once more our interpretation of the observed values, by considering the set *instMatching(P)* for a property *P*, introduced with the help of an example as follows.

Consider properties *edition* and *rating*. Replace the observed values by *ordered pairs* of the form *(instanceID,value)*, where the *instanceID* is the Uniform Resource Identifier of the instance and *value* is the value of the property. Replace the *instanceIDs* of target pairs by *instanceIDs* from the source, when there is an equivalent instance from the source. The characteristic sets thus obtained are shown in Figure 8. By doing so, the induced matchings are correct, since there are no false positives, because the sets of pairs are completely different for `source:edition` versus `target:rating` and `source:rating` versus `target:edition`. We then compute $\sigma(instMatching[U_i], instMatching[V_j])$.

source:isbn, target:isbn → 60% commonalities
(3 common isbn in a total of 5 different titles)

source:title, target:title → 60% commonalities
(3 common titles in a total of 5 different titles)

source:edition, target:edition → 100% commonalities

source:rating, target:rating → 100% commonalities

source:edition, target:rating → 100% commonalities

source:rating, target:edition → 100% commonalities

Figure 7. Induced matchings corresponding to Figure 6, using the observed values/tokens heuristic.

Note that the string domain heuristic can be used in conjunction with the instance matching heuristic for string properties. Instead of *(instanceID,value)* pairs we would use *(instanceID, token)* pairs.

Let A_i be a property of C. Let μ_I an instance matching between C and D. The *observed instance-value pairs representation of* A_i in $\boldsymbol{C}$ is the set $\boldsymbol{iv}[\boldsymbol{C},A_i]$ such that $(I,v) \in \boldsymbol{iv}[\boldsymbol{C},A_i]$ iff there is a triple of the form (I,A_i,v) in $\boldsymbol{C}$. The *observed instance-value pairs representation of* B_j in $\boldsymbol{D}$ is the set $\boldsymbol{iv}[\boldsymbol{D},B_j]$ such that $(I,w) \in \boldsymbol{iv}[\boldsymbol{D},B_j]$ iff there is a triple of the form (J,B_j,w) in $\boldsymbol{D}$ and a pair $(I,J) \in \mu_I$. Note that this definition is not symmetric with respect to source and target.

(i) First step: replacement of the observed values by instance-value pairs

source:edition ∈ {(10,1), (20,2), (30,3),(40, 4)}
source:rating ∈ {(40,1), (10,2), (20,3),(30, 4)}

target:edition ∈ {(100,1), (200,2), (300,3), (400,4)}
target:rating ∈ {(400,1), (100,2), (200,3), (300,3)}

(ii) Second step: replacement of the target instanceID's by source instanceID's

target:edition ∈ {(10,1), (20,2), (300,3), (40,4)}
target:rating ∈ {(40,1), (10,2), (20,3), (300,3)}

Figure 8. Representation of the properties *edition* and *rating* of Figure 6 with instance matching heuristic.

Let ***IV*** denote the universe of *(instanceID,value)* pairs. Consider a (generic) similarity function $\sigma: 2^{IV} \times 2^{IV} \rightarrow \mathbb{R}^+$ over 2^{IV} and a positive Real number, τ, the *similarity threshold* for sets of values in ***IV***. We define the *property matching between C and D, based on instance-value pairs and induced by* σ *and* τ, as the partial, many-to-many relation

$\mu_P \subseteq \{A_1,...,A_m\} \times \{B_1,...,B_n\}$ such that $(A_i,B_j) \in \mu_A$ iff $\sigma(\mathbf{iv}[\mathbf{C},A_i],\mathbf{iv}[\mathbf{D},B_j]) \geq \tau$.

To sum up, at this point we have defined two characteristic sets for a property *P* of a given catalogue *C*, ***o[C,P]*** and ***iv[C,P]***, and one characteristic set for thesauri terms, $\mathbf{i}[\mathbf{U}_T,t]$. Generically, we refer to a characteristic set as *featureSet[C,P]*.

The fourth heuristic, called the *type compatibility heuristic*, is quite simple: we compute the similarity between two properties only if they are of the same type (or of compatible types). This heuristic is advantageous since it avoids testing all $(m \times n)/2$ possible combinations of properties from *C* with properties from *D*, assuming that the similarity function is symmetric (note that it is not when we consider observed instance-value pairs representations).

The fifth heuristic, called the *multiset domain heuristic*, is to consider the *multiset of observed values* of a property *P*, defined as the multiset that contains as many elements corresponding to a single value *v* as the number of triples whose value for *P* is *v*. Intuitively, the motivation for this heuristic is to take into account, in the similarity function, the number of times a value occurs for a property. The results in Section 3.5 suggest that this heuristic is actually not very effective for property matching.

3.4. Formalization of catalogue matching

In this section, we introduce the notion of similarity-induced catalogue matching based on the similarity heuristics of the previous section.

Let $C[A_1\ U_1, ..., A_m\ U_m]$, with thesaurus *T*, and $D[B_1\ V_1, ..., B_n\ V_n]$, with thesaurus *W*, be the *source* and the *target catalogue schemas*.

Let ***C*** and ***D*** be extensions of source and target catalogues.

A *similarity-based matching model* for catalogues consists of:

- a *similarity-based instance matching* μ_I, which is a possibly many-to-many relationship between pairs of instance representations
- a *similarity-based term matching* μ_T, which is a possibly many-to-many relationship between pairs of term representations
- a *similarity-based property matching* μ_P, which is a possibly many-to-many relationship between pairs of property representations

Defining a similarity-based matching model requires addressing three problems:

1. Deciding on how to represent the objects – instances, properties, thesauri terms – to be matched;
2. Defining a similarity measure that applies to the selected representations; and
3. Based on the similarity measure of their representations, deciding when two objects match.

For the sake of concreteness, we adopt as similarity measure variations of the estimated mutual information matrix.

Let $A=(A_1,...,A_m)$ and $B=(B_1,...,B_n)$ be two lists of sets. Recall from Section 2.2 that the *estimated mutual information matrix* for A and B is the $m \times n$ matrix EMI such that, for each $r \in [1,m]$ and $s \in [1,n]$:

$$(1) \qquad EMI_{rs} = \frac{m_{rs}}{M} \log\left(M \frac{m_{rs}}{\sum_j m_{rj} * \sum_i m_{is}} \right)$$

where $m_{ij} = |\sigma_R[A_i] \cap \sigma_S[B_j]|$, for $i \in [1,m]$ and $j \in [1,n]$, and $M = \sum_{i,j} m_{ij}$.

Observe that the EMI matrix is therefore symmetric, since $|\sigma_R[A_i] \cap \sigma_S[B_j]| = |\sigma_R[A_i] \cap \sigma_S[B_j]|$. We also say that $[m_{ij}]$ is the *co-occurrence matrix* of A and B.

Consider the problem of applying the estimated mutual information matrix to match thesauri terms. Assume that we have already defined an instance matching $\mu_I \subseteq C \times D$ for these catalogues. Recall that μ_I is a possibly many-to-many relationship between instances in C and instances in D. Also recall that we say that an instance I in C matches an instance J in D iff $(I,J) \in \mu_I$. Finally, recall that the *representation* of a term t of T in C is the set $\boldsymbol{i}[\boldsymbol{C},t]$ of all instances in $\boldsymbol{C}$

whose type is ***t***. Likewise, the *representation* of a term u of W in D is the set $\boldsymbol{i}[\boldsymbol{D},u]$ of all instances in $\boldsymbol{D}$ whose type is u. This settles the first problem for thesauri terms.

To apply the concept of estimated mutual information matrix, first consider that the terms in T and W are arbitrarily ordered as $t_1,...,t_m$ and $u_1,...,u_n$, respectively. We cannot directly compute the estimated mutual information matrix for $\boldsymbol{i}[\boldsymbol{C},T]=(\boldsymbol{i}[\boldsymbol{C},t_1],...,\boldsymbol{i}[\boldsymbol{C},t_m])$ and $\boldsymbol{i}[D,W]=(\boldsymbol{i}[\boldsymbol{D},u_1],...,\boldsymbol{i}[\boldsymbol{D},u_n])$ since $\boldsymbol{i}[\boldsymbol{C},t_i]$ and $\boldsymbol{i}[\boldsymbol{D},u_j]$ are heterogeneous sets, that is, we cannot directly compute the cardinality of their intersections. However, we may redefine m_{ij} to be the cardinality of the matching set between instances in $\boldsymbol{i}[\boldsymbol{C},t_i]$ and instances in $\boldsymbol{i}[\boldsymbol{D},u_j]$, that is, the cardinality of $\mu_I \cap \boldsymbol{i}[\boldsymbol{C},t_i] \times \boldsymbol{i}[\boldsymbol{D},u_j]$. With this proviso, we may compute the estimated mutual information matrix between terms of T and terms of W, which settles the second problem for thesauri terms.

Note that, since μ_I is not necessarily one-to-one, the number of instances of $\boldsymbol{i}[\boldsymbol{C},t_i]$ that match instances in $\boldsymbol{i}[\boldsymbol{D},u_j]$ is not necessarily equal to the number of instances of $\boldsymbol{i}[\boldsymbol{D},u_j]$ that match instances in $\boldsymbol{i}[\boldsymbol{C},t_i]$. Therefore, to avoid this asymmetry, we decided to define m_{ij} to be the cardinality of $\mu_I \cap \boldsymbol{i}[\boldsymbol{C},t_i] \times \boldsymbol{i}[\boldsymbol{D},u_j]$.

As for the third problem, there are two directions to follow. Given the estimated mutual information matrix *EMI* between terms of T and terms of W, we may decide that two terms t_r and u_s match iff EMI_{rs} is the largest entry column wise and row wise, that is, we may define a thesauri matching μ_T between terms of T and terms of W as follows:

(2) $(t_r,u_s) \in \mu T$ *iff* $EMI_{rs} \geq EMI_{rj}$, for all $j \in [1,n]$, *with* $j \neq s$, and $EMI_{rs} \geq EMIis$, for all $i \in [1,m]$, *with* $i \neq r$

for each $r \in [1,m]$ and $s \in [1,n]$

We say that this thesauri matching is *directly derived* from the estimated mutual information matrix. Note that Equation (2) induces a one-to-one thesauri matching, except when there are two entries, EMI_{rs} and EMI_{vw}, such that $EMI_{rs} = EMI_{vw}$ and both satisfy Equation (2). To force Equation (2) to induce one-to-one matchings, we arbitrarily take the smallest r and the smallest s when there is a tie.

Alternatively, we may define that two terms t_r and u_s match iff EMI_{rs} is

above a certain threshold τ_T:

(3) $(t_r,u_s) \in \mu_T$ *iff* $EMI_{rs} \geq \tau_T$, for each $r \in [1,m]$ and $s \in [1,n]$

which induces a possibly many-to-many thesauri matching. We say that this thesauri matching is *derived* from the estimated mutual information matrix *with the help of the threshold* τ_T. This second approach requires experimentation to decide on the threshold value, but it has the advantage of accounting for potentially non one-to-one matchings.

Let us now move to the problem of applying the estimated mutual information matrix to match properties. Deciding on how to represent properties is a problem open to several alternative solutions.

Let A_i be a property of C. Recall from Section 3.3 that the observed domain representation of A_i in C is the set $\boldsymbol{o}[\boldsymbol{C},A_i]$. Also recall that we redefined such set to consider property values and tokens derived from string property values. The observed domain representation of a property of D is likewise defined. We then compute the estimated mutual information matrix for the lists of sets $\boldsymbol{o}[\boldsymbol{C},A]=(\boldsymbol{o}[\boldsymbol{C},A_1],...,\boldsymbol{o}[\boldsymbol{C},A_m])$ and $\boldsymbol{o}[\boldsymbol{D},B]=(\boldsymbol{o}[\boldsymbol{D},B_1],...,\boldsymbol{o}[\boldsymbol{D},B_n])$. We may improve the construction of the matrix by computing m_{ij} using the *type compatibility heuristic*.

Finally, recall from Section 3.3 that the multiset instance matching representation of a property is the set $\boldsymbol{iv}[\boldsymbol{C},A_i]$. We then compute the estimated mutual information matrix for the lists of sets $\boldsymbol{iv}[\boldsymbol{C},A]=(\boldsymbol{iv}[\boldsymbol{C},A_1],...,\boldsymbol{iv}[\boldsymbol{C},A_m])$ and $\boldsymbol{iv}[\boldsymbol{D},B]=(\boldsymbol{iv}[\boldsymbol{D},B_1],...,\boldsymbol{iv}[\boldsymbol{D},B_n])$.

Given the estimated mutual information matrix *EMI* between properties of C and properties of D, we may derive a property matching μ_P between these properties as for thesauri terms, using equations (2) or (3). However, we have to decide on a particular representation for the properties. We may in fact go further and: (1) use several different representations, thereby generating several matrices, $EMI^1,..., EMI^k$; (2) compute the final matrix *EMI* by combining $EMI^1,..., EMI^k$ in a specific way, such as by taking EMI_{rs} as the maximum of $EMI^1_{rs},..., EMI^k_{rs}$; (3) compute the property matching from the final matrix *EMI*, using equations (2) or (3).

Finally, we briefly comment on how to match instances from $\boldsymbol{C}$ and $\boldsymbol{D}$. We

consider that a catalogue instance is *represented* by a list of some of its property values, that is, we admit that some properties may be left out of the instance representation.

Let $L = (a_{i_1},...,a_{i_p})$ be a representation of an instance *I* from ***C***, using the values of properties $A_{i_1},...,A_{i_p}$, and $M = (b_{j_1},...,b_{j_q})$ be a representation of an instance *J* from ***D***, using properties $B_{j_1},...,B_{j_q}$. Assume, for the sake of argument, that $p \leq q$. Assume also that we have an one-to-one property matching μ_P between properties of *C* and properties of *D* that cover all properties in $A_{i_1},...,A_{i_p}$. Let *J'* be a permutation of *M*, truncated up to the p^{th} entry, such that now property A_{i_r} matches property B_{j_r}, according to μ_P, for $r \in [1,p]$. Then, adopting a strategy similar to that described above for thesauri terms and properties, we may derive an instance matching from any vector similarity measure applied to *I* and *J'*, such as the cosine distance (see Section 2.2).

For example, in Section 3.2, we represented a geographic object by its position and name, and considered that two instances from the different gazetteers match if their geographic position and name are similar, using cosine distance. A simpler example would be to consider that instances from different book catalogues are represented by their ISBN properties, and that they match iff they have the same ISBN values.

Note that the instance matching defined above depends on a property matching and on a correct interpretation of the properties, which may be informed by the user or, in simple cases, inferred by the system. Furthermore note that both the co-occurrence matrix for thesauri terms and the instance matching representation for properties require that an instance matching be defined, which in turn depends on a property matching. In other words, such concepts are not orthogonal and require a careful engineering to avoid circularities. The examples in Section 3.5 indeed start with very simple instance matchings to derive thesauri and property matchings. In Chapter 4 we will tackle this problem in depth and suggest a generic technique for instance matching.

3.5. Experiments

3.5.1. Data sources

We conducted two experiments to assess the performance of several similarity-based matching models. The first experiment was based on data extracted from the GEOnet Names Server (GNS) and the Alexandria Digital Library gazetteer (ADL), already used in Section 3.2. The second experiment was based on data about books obtained from Amazon and Barnes & Noble. All these data sources provide Web service access, except Barnes & Noble, in which case we developed an HTML parser to capture data from query results.

For each experiment, we first defined a bootstrap set of keywords, which we used to query the databases. From the query results, we extracted the less frequent words, from 1 to 10 occurrences. Since the keywords occur just a few times in the query results, it is expected that each keyword identifies small sets of database entries and, therefore, they can be used to query the other database to find the same products. This pre-processing step enhanced the probability of retrieving duplicate objects from the databases, which is essential to evaluate any extensional schema matching technique. For the first experiment, we extracted a total of 23,390 records: 3,599 from GNS and 19,791 from ADL. For the second experiment, we extracted a total of 116,201 records: 16,410 from Amazon and 99,791 from Barnes & Noble.

3.5.2. Experiments with gazetteers

3.5.2.1. Thesauri matching

The experiments described in this section focused on matching the ADL Feature Type Thesaurus (FTT) with the GEOnet Names Server classification scheme (GNS CS). Although the ADL FTT has a total of 1,262 terms, we considered only the preferred terms, which amounts to 210 terms. The GNS CS has 642 terms, organized into two levels, with 9 top terms.

The experiments had the following characteristics:

1) Used the data extracted from ADL and GNS, as described in Section 3.5.1.
2) Adopted a simple instance matching, computed using the centroids and the names of the geographic features.
3) Tested the thesauri matching model directly derived from the estimated mutual information matrix, with each thesauri term *t* represented as the set of all instances whose type is *t*.

The instance matching adopted assumes that: (1) a geographic feature F_i is represented by the triple $(long_i, lat_i, N_i)$, where $(long_i, lat_i)$ is the centroid and N_i is the name of the feature; (2) a matching between the properties of ADL and GNS that store the centroid and the name of a geographic feature has been defined. These assumptions avoid the circularity problem mentioned at the end of Section 3.4, for the sake of simplicity and clarity of the experiment.

We then define that two geographical features match iff their centroids and their names match, computed as follows. Let F_i and F_j be two features. Then, we considered that their centroids match iff

$$\sqrt{(long_j - long_i)^2 + (lat_j - lat_i)^2} \leq 0.9$$

To compare N_i and N_j, we first computed the vector similarity *v* between the token vectors built from N_i and N_j, taking the TF-IDF weight for each token. Then, we considered that N_i and N_j match iff $v \geq 0.9$.

In order to evaluate the technique we used the performance measures of *precision*, *recall* and *overall performance*, denoted simply as *f*. These measures are defined as follows according to (Manning and Schütze 2002).

$$precision = \frac{true.positive}{true.positive + false.negative}$$

$$recall = \frac{true.positive}{true.positive + false.negative}$$

$$f = 2 * \frac{precision * recall}{precision + recall}$$

Furthermore note that the thesauri matching model directly derived from the

estimated mutual information matrix is one-to-one, by definition. Therefore, for fairness, the reference thesauri matching must contain only one-to-one matchings.

Table 2 shows a fragment of the co-occurrence matrix and Table 3, the corresponding EMI matrix. The highlighted cells have the largest values of their respective rows and columns. Table 4 contains the thesauri matchings directly

Table 2. A fragment of the co-occurrence matrix for ADL and GNS.

	ADL									
GNS	islands	lakes	mountains	populated places	railroad features	reference locations	ridges	rivers	streams	waterfalls
FLLS			1	10					5	353
FRM				44		1			13	
HLL		2	177	14	1	1			5	
HLLS			136	27			2		24	
INLT				6					2	
ISL	460		1	39		3	1		18	
ISLS	20			3						
LCTY	2		7	37			2		13	
LGN		62	1	11	1				5	
LK		310	1	7	1		3		5	
LKI		2								
LKO		10								
LKS		2								
MT			74	7			3		4	
MTS			68	22	2		3		14	
PPL	32	23	83	7440	52	24	30	2	799	13
PPLA			1	6					2	
PPLL				6						
PPLX		1	1	28	2					
PS				1						
PT	3			34		181			2	
RDGE	1	1	4	21	3		101		19	1
RSTN		2	1	30	300	1	2		21	
RSTP			1	18	141		1		12	
RSV				1						
SCH				1						
SCRP			1	1					1	
SPUR		2		5			32		5	
STM	21	10	58	667	28	2	31		4732	10
STMI	2		2	90	2		2		251	

Table 3. EMI matrix corresponding to the matrix in Table 2.

	ADL								
GNS	islands	lakes	mountains	Populated places	Railroad features	Reference locations	ridges	streams	waterfalls
FLLS									0.032134
HLL			0.013706						
HLLS			0.009877						
ISL	0.037034								
ISLS	0.001607								
LCTY	0.000004		0.000200	0.000208			0.000048		
LGN		0.005162							
LK		0.027295							
LKI		0.000179							
LKO		0.000893							
LKS		0.000179							
MT			0.005609				0.000075		
MTS			0.004708				0.000061		
PPL				0.108805					
PPLA			0.000028	0.000049					
PPLL				0.000107					
PPLX		0.000007		0.000412	0.000036				
PT						0.018135			
RDGE							0.009716		
RSTN					0.023947				
RSTP					0.011171				
SCRP			0.000028						
SPUR		0.000031					0.003147		
STM								0.107120	
STMI								0.004676	

derived from the EMI matrix, according to Equation (2). The complete analysis of the results indicates a total of 41 true positive matchings over a total of 43 correct matchings (true positives + false negatives), which means a *recall* of 95%. By contrast, it indicates a total of 9 false positive matchings, which means a *precision* of 88%. The *overall performance* is then $f = 2*0.88*0.95/(0.95+0.88)=91\%$

Table 4. Matchings directly derived from the EMI matrix of Table 2.

Matchings		
ADL	GNS	type
islands	ISL	tp
lakes	LK	tp
mountains	HLL	tp
populated places	PPL	tp
railroad features	RSTN	tp
reference locations	PT	tp
ridges	RDGE	tp
streams	STM	tp
waterfalls	FLLS	tp
bays	*BCH*	*fp*
canals	*STMC*	*fp*

3.5.2.2. Property matching

The experiments described in this section concentrated on matching the ADL gazetteer property list, shown in Table 5, with the GNS property list, shown in Table 7. The experiments had the following characteristics:

1. Used the data extracted form ADL and GNS, as described in section

Table 5. ADL property list.

Property	Description	Datatype
boundingBoxX1	longitude of the left upper corner of the bounding box containing the feature	Real
boundingBoxY1	latitude of the upper left corner of the bounding box containing the feature	Real
boundingBoxX2	Longitude of the lower right corner of the bounding box containing the feature	Real
boundingBoxY2	Latitude of the lower right corner of the bounding box containing the feature	Real
displayName	display name	String
footprintX	longitude of the centroid of the bounding box of the location of the object	String
footprintY	latitude of the centroid of the bounding box of the location of the object	String
identifier	entry local id	String
names	alternative names	String
placeStatus	entry place-status (current or former)	String
relationships	Relationships with other features	String

3.5.1.

2. Adopted a simple instance matching, computed using the centroids and the names of the instances, as in Section 3.5.2.1.
3. Tested the family of property matching models directly derived from the estimated mutual information matrix, with each property represented as described in section 3.4 (all models adopt the type compatibility heuristics).

Table 7. GNS property list.

Property	Description	Datatype
adminCode1	Code for 1st administrative division	String
adminName1	Name for 1st administrative division	String
alternateNames	alternative names	String
countryCode	country code (ISO-3166 2-letter code)	String
countryName	country name	String
elevation	elevation, in meters	Real
geonameId	identifier	String
lat	latitude of the centroid of the bounding box of the location of the object	Real
lng	longitude of the centroid of the bounding box of the location of the object	Real
name	primary name	String
population	population	Integer

In this experiment, we also used the same performance measures; *precision*, *recall* and *overall performance (f)*; for evaluating the matching approach. All measures were computed taking into account the reference property matchings of Table 6.

Table 8 shows the performance results for the property matching models directly derived from the estimated mutual information matrices computed using different property representations and combinations thereof. For the first model in

Table 6. Reference property matchings for ADL and GNS.

ADL	GNS
displayName	name
footprintX	lng
footprintY	lat
names	alternateNames

Table 8, we show in Table 9, Table 10 and Table 11 the corresponding co-occurrence matrix, the estimated mutual information matrix and the directly derived property matchings. The complete analysis of the results for the first model indicates a total of 4 *true positive* matchings over the total of 4 correct alignments (*true positives* + *false negatives*), which means a recall of *100%* of the total corrects matchings. By contrast, it indicates 1 *false positive* matching (cell with dashed line in Table 10), which means a precision of *80%*. The *overall performance (f)* is then 89%.

Table 8. Performance of the property matching models directly derived from the EMI matrix.

instance matching	observed domain	multiset	type compatibility	precision	recall	*f*
false	*true*	*false*	*true*	80%	100%	89%
false	*true*	*true*	*true*	50%	50%	50%
true	*false*	*false*	*true*	50%	50%	50%
true	*false*	*true*	*true*	50%	50%	50%
true	*true*	*false*	*true*	50%	50%	50%
true	*true*	*true*	*true*	50%	50%	50%

Notes:

1) The first two columns indicate the property representation that the model adopts (instance matching or observed domain representations).
2) For each line, when the value of the multiset column is *True*, it indicates that the idea of using a multiset is applied to both the instance matching and the observed domain representations.
3) The type compatibility column is all *True*, indicating that all models use the type compatibility heuristics.
4) The last two lines, where both the instance matching and the observed domain columns are *True*, correspond to the models based on an EMI matrix obtained by taking, for each entry, the maximum value from the EMI matrix computed using the instance matching representation and the EMI matrix computed using the domain value representation.

Note: we disregarded properties *boundingBoxX1*, *boundingBoxY1*, *boundingBoxX2*, *boundingBoxY2* since they actually contain the same values as *footprintX*, *footprintY* in the sample data downloaded from ADL.

Table 9. A fragment of the co-occurrence matrix for properties of ADL and GEOnet.

GNS	ADL				
	displayname	footprintX	footprintY	names	relationships
admincode1	23	9	15	25	11
adminname1	156			110	160
alternateNames	252		1	371	59
countryCode	4			4	4
countryName	59			21	62
elevation	4	2	1	4	
lat	8	222	1250	8	
lng	2	1323	445	2	
name	381			382	43
population					1

Table 10. EMI matrix corresponding to the co-occurrence matrix in Table 9.

GNS	ADL				
	displayname	footprintX	footprintY	names	relationships
admincode1	0.00086	0.00008	0.00025	0.00095	0.00046
adminname1	0.00666			0.00387	0.00983
alternateNames	0.01079			0.01832	0.00197
countryCode	0.00016			0.00016	0.00024
countryName	0.00266			0.00052	0.00399
elevation	0.00017	0.00004	0.00000	0.00017	
lat		0.00331	0.05750		
lng		0.06029	0.01023		
name	0.01811			0.01787	0.00104
population					0.00008

Table 11. Property matchings corresponding to the third model in Table 10.

ADL	GNS	Type
footprintX	lng	tp
footprintY	lat	tp
names	*alternateNames*	*fp*
relationships	adminName1	tp
displayName	Name	tp

3.5.3. Experiments with book catalogues

The experiments described in this section repeat those of Section 3.5.2 for the Amazon and the Barnes & Noble book catalogues. Table 12 and Table 13 show the Amazon and the Barnes & Noble property lists, whereas Table 14 contains the reference property matchings.

In this experiment, we assumed that: (1) an instance from Amazon is represented by the values of properties *title*, *author*, *publisher* and *isbn*; (2) an instance from Barnes & Noble is represented by the values of properties *name*, *by*, *publ* and *isbn-13*; (3) the properties in these two lists match (see however the observation about *isbn-13* below). We considered that two instances *match* iff their representations are similar, using as similarity measure the cosine distance with TF-IDF, and a threshold of 0.9.

Table 12. Amazon property list.

Property name	Description	Datatype
author		String
edition		Integer
index	Book classification	String
isbn		String
listPrice		Real
productGroup		String
productType		String
publisher		String
title		String
url		URL

Table 13. Barnes & Noble property list.

Property name	Description	Datatype
by	author	String
category	book classification	String
isbn-13	the 13-digit International Standard Book Number	Integer
name	title of the book	String
numberOfPages	number of pages	Integer
pubDate	publication date	Date
publ	Publisher	String
salesRank	number of times that other titles sold more than this book title	Integer
subject		String

Table 14. Reference property matchings for the Amazon and Barnes & Noble book catalogues.

Amazon	Barnes & Noble
author	by
index	category
publisher	publ
title	name

Table 15 shows performance results for the property matching models directly derived from the estimated mutual information matrix. It should be interpreted as Table 8. Table 15 indicates that the matching models based on the instance matching representation for properties (lines 3 to 6) do not have the best performance. This can be explained in part since, in this sample data, the number of instances from both catalogues that match is fairly low. For the first model in Table 15, Table 17 and Table 16 show the occurrence and the estimated mutual information matrices computed, and Table 18 shows the property matchings derived.

Table 15. Performance results for the property matching models directly derived from the EMI matrix.

instance matching	observed domain	multiset	type compatibility	precision	recall	*f*
false	*true*	*false*	*true*	80%	100%	89%
false	*true*	*true*	*true*	100%	75%	86%
true	*false*	*false*	*true*	57%	100%	73%
true	*false*	*true*	*true*	57%	100%	73%
true	*true*	*false*	*true*	57%	100%	73%
true	*true*	*true*	*true*	60%	75%	67%

An interesting observation can be made regarding ISBN values. Starting in 2007, the 13-digit ISBN began to replace the 10-digit ISBN. The Amazon book catalogue stores both numbers, with the property *isbn* holding the old 10-digit ISBN and the property *ean* (not used in the experiment), the new 13-digit ISBN. The Barnes & Noble book catalogue stores only the new 13-digit ISBN (the property *isbn-13*). Differently from a syntactical approach, which would wrongly match *isbn* with *isbn-13*, due to their syntactical similarity, our instance-based technique did not match *isbn* with *isbn-13*, since obviously these properties have no common values (they are in fact omitted from Table 16 and Table 17).

The date properties also never matched due to differences in format. Indeed, Amazon stores dates in the format “YYYY-MM-DD”, while the Barnes & Noble stores the publication date as “Month, YEAR”. To solve this problem, we would have to consider a more sophisticated strategy to compare dates.

Table 16. A fragment of the co-occurrence matrix for properties of Amazon and Barnes & Noble corresponding to the first model of Table 15.

	Barnes & Noble							
Amazon	by	category	name	numberOfPages	pubDate	publ	salesRank	subject
author	2580	1	927	3	3	377	2	26
edition	60	1	137	23	22	49	29	2
index	1	1	1			1		1
label	3		62	148	12	1	166	
listprice	5	1	12			6		4
productGroup	913	1	1138	12	20	890	10	48
productType	1642	1	3785	149	77	761	159	62
publisher			3				1	
title	2580	1	927	3	3	377	2	26
url	60	1	137	23	22	49	29	2

Table 17. EMI matrix corresponding to the co-occurrence matrix in Table 17.

	Barnes & Noble							
Amazon	by	category	name	numberOfPages	pubDate	publ	salesRank	subject
author	0.018669							
edition	0.000422			0.002287	0.001848		0.000885	
index		0.014983				0.001488		0.012317
listPrice				0.005492	0.000726		0.004622	
productGroup		0.014797				0.001495		0.012277
productType	0.000001							0.000001
publisher	0.005598					0.014014		
title			0.015650	0.000947	0.000751			
url			0.029205				0.024320	

Table 18. Property matchings corresponding to the first model in Table 15.

Amazon	Barnes & Noble	
author	by	tp
index	category	tp
publisher	publ	tp
title	name	tp
listPrice	*numberOfPages*	*fp*

3.6. Summary and contributions

In Chapter 3, we proposed an approach to match pairs of catalogues. This problem was chosen as the starting point for introducing the concepts of a general schema matching technique because: (1) catalogues have simple schemas; (2) catalogues are a recurrent data source in e-business; and (3) current techniques for catalogue matching do not take into account the thesauri matching subproblem.

The approach is classified as extensional since it uses instances stored in the catalogues, and is based on the notion of similarity. To provide the foundations of the discussion, we first defined the concepts of thesauri, property and instance matchings, and discussed how to use similarity functions to induce matchings. Specifically, we adopted the estimated mutual information (EMI) matrix to measure similarity and defined how to derive thesauri and property matchings from the EMI matrix. We also called attention to the fact that properties may have alternative representations, which impact the computation of the EMI matrix. Finally, we illustrated the approach with experiments using data from catalogues available on the Web. The experiments also measured the influence of the alternative property representations on the performance of the property matchings derived.

The results described admit at least three extensions, as described in (Leme et al. 2008b). First, although we concentrated on just two catalogues, we may extend the overall approach to match multiple catalogues by computing the EMI matrix between any two catalogues. Second, in addition to one-to-one matchings, we may derive many-to-many matchings by using the EMI matrix as in equation (3), as well as by adopting other similarity functions. The results are still promising, but they require a training step to calibrate the threshold value (of

equation (3)), and additional parameters, when other similarity functions are adopted (Leme et al. 2008b). Finally, we have not discussed how to gradually construct the matchings as new data from the catalogues are available, which is typical of a query mediation environment. We refer the reader to (Brauner et al. 2006, Brauner et al. 2008) for discussions about this issue.

Leme et al. (Leme et al. 2008b) show that the co-occurrence matrix (EMI) approach is not the similarity model with the best overall performance (best *f*). The Contrast Model (CM) proved to be more efficient in detecting matching elements, but it requires a training process. The co-occurrence matrix, on the other hand, can be used without this expensive process and performed fairly well in the experiments.