

Utility Ontology Development with Formal Concept Analysis

Gaihua Fu and Anthony G Cohn
School of Computing, University of Leeds, UK
{gaihua, agc}@comp.leeds.ac.uk

Abstract. Although it is well recognised that ontologies have an important role to play in data integration, the lack of established ontologies in domains of interest often makes ontology-based integration a difficult task. Previous research on ontology design methodologies shows that manual construction of ontologies is a complex process and it is very hard for a designer to develop a consistent ontology. This paper contributes a formal and semi-automated approach for the development of ontologies in the utility infrastructure domain. It arises from a practical industrial problem of integrating the vast network of underground asset records. These asset records are typically autonomous, i.e. owned and maintained by individual organisations, and are encoded in an uncoordinated way, i.e. without consideration of interoperability with other utility information systems. The proposed approach is based on formal concept analysis (FCA) which is a mathematical approach for abstracting from attribute-based object descriptions. This paper describes techniques developed to support utility ontology development, with a focus on resolving implicit and mismatch data. Some experiments have been carried out to construct a utility ontology with data from utility companies. Though issues addressed in the paper arise in utility ontology development, we anticipate that they should be interesting and relevant to other application domains.

Keywords: Ontology, Integration, Formal Concept Analysis

1. INTRODUCTION

In many domains, one faces the need for exchanging and sharing information that comes from different resources. Obtaining mapping information of the excavation site in street works is one such example. Every year, in excess of four million holes are dug in UK roads to maintain utility assets [2]. In order to avoid unnecessary holes dug in wrong places, it is required that information of buried utilities must be obtained before excavation occurs. However, the mapping information supplied by utilities is often of limited use. One main reason for this is that asset records are usually created and maintained by a range of private companies with little thought towards interoperability. As a result, utility data differs from one company to another not only in *what* is encoded but also *how* it is encoded. This data heterogeneity makes it extremely difficult for the excavators to synthesize an integrated view of the excavation site.

Overcoming data heterogeneity has been an active area of research in database and information integration communities [3, 14]. Ontology research is another discipline that deals with data heterogeneity [18, 12]. The common definition of the term ontology is that an ontology is some formal, explicit specification of a domain of

discourse. Ontology-based integration systems are usually characterised by a global ontology which represents a reconciled, integrated view of the underlying data sources. Systems taking this approach usually provide users with a uniform interface – all queries made to source data are expressed in terms of the global ontology, thus freeing them from the need to understand each individual data source. This approach looks straightforward but a shared ontology is required. Unfortunately, in many domains one faces the problems of having no accepted ontologies that can be employed in the integration work. The utility domain is one example of this. According to our investigations, little work has been attempted previously to develop utility ontologies. The most relevant one is the standard proposed by the FGDC [4]. However, our research [7] show that the knowledge encoded in FGDC standard is insufficient to serve as a reference ontology on which to base UK utility data integration.

To attempt to remedy this, we investigate techniques that support ontology development for utility domain. In [7], we reported on how a basic ontology was developed manually for the water utility domain. Our experiences show that manual construction ontology is a time-consuming process and it is very hard for a designer to develop a consistent ontology. In this paper we present an alternative approach for utility ontology development. The method is based on formal concept analysis (FCA). By deriving conceptual structures based on attribute descriptions of utility asset features, the method supports the development of utility ontologies in a systematic and semi-automated manner. We discuss interesting issues when applying FCA in ontology development, particularly the treatment of implicit and ambiguous information. We report on our experiments on employing the proposed techniques to construct a sewer utility ontology.

The remaining part of the paper is organized as follows. Section 2 recalls key notions of FCA and reviews related research. Section 3 presents our method for utility ontology development based on FCA. Sections 4 and 5 discuss techniques that deal with implicit and ambiguous information. Section 6 reports our experimental results. Section 6 concludes the paper and points out future research.

2. Basic Concepts and Related Research

2.1. Formal Concept Analysis (FCA)

FCA was developed in the 1980s [26]. A typical task that FCA can perform is data analysis, making the conceptual structure of the data visible and accessible.

A basic notion for FCA is a formal context, which is defined as a triple $K := \langle G, M, I \rangle$, where G is a set of objects, M is a set of attributes, and $I \subseteq G \times M$ is a binary relation between G and M . A relation $(g, m) \in I$ is read as “object g has the attribute M ”. A formal context can be depicted by a cross table as shown on the left side of Figure 1, where the elements on the left side are objects; the elements at the top are attributes; and the relation between them is represented by the crosses.

A formal concept of a context $K := \langle G, M, I \rangle$ is defined as pair (A, B) , where $A \subseteq G$, $B \subseteq M$, $A' = B$ and $B' = A$. A' is the set of attributes common to all the objects in A and B' is the set of objects having the attributes in B . The *extent* of the concept (A, B) is A and its *intent* is B . The formal concepts of a context are ordered by the *sub-* and *super-*concept relation. The set of all formal concepts ordered by such relations forms a

concept lattice. For example, the right side of Figure 1 shows the classic concept lattice corresponds to the context on the left side of Figure 1.

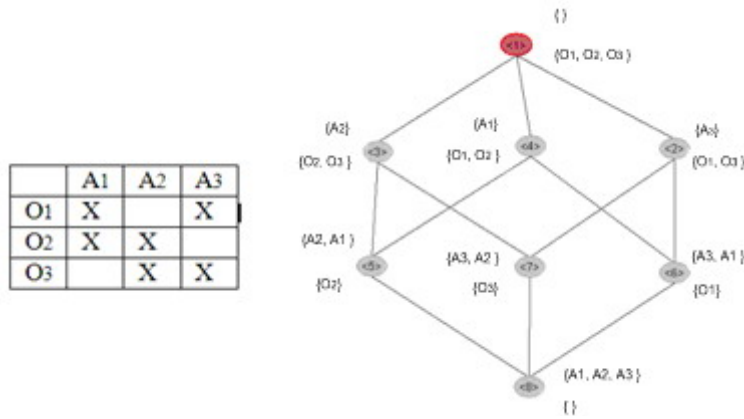


Figure 1 A Formal Context and its Corresponding Concept Lattice

The formal contexts we introduced above are not the ones that occur most frequently in applications of FCA. Most often data is encoded in form of *many valued contexts*. A many valued context $K := \langle G, M, W, I \rangle$ consists of a set of objects G , a set of attributes M , a set of attribute values W , and a ternary relation $I \subseteq G \times M \times W$. A relation $(g, m, w) \in I$ is read as “object g has the attribute M and its value is w ”. A many valued context can be unfolded into a one valued context through conceptual scaling (the reader is referred to [6] for discussion on conceptual scaling).

2.2. Related Research

A growing number of methods have been proposed in recent years to address the issues of ontology development [12, 10]. Most methods are based on the traditional knowledge engineering approach [5, 16, 22, 23]. These methods usually start with defining the domain and scope of ontologies. This is followed by a data acquisition process: important classes are collected; a class hierarchy is derived; and properties and semantic constraints are defined and attached to classes. The next step is the specification of ontologies in some formalisms. Finally, ontology evaluation is performed against pre-defined criteria. It is recognised that developing ontologies from scratch with the traditional knowledge engineering approach is time consuming and labour intensive. Reusing and integrating existing ontologies is thus considered as part of ontology development process by most methods described above. However few of these methods address integration in detail. In [20], a method that is based on the integration is proposed. Other systems and frameworks that have been developed for supporting ontology merging or integration tasks include those described in [8, 17].

Fundamental operations for ontology integration are mapping discovery and ontology merging. Mapping discovery takes two or more ontologies as input and produces a mapping between elements that semantically correspond to each other. Most approaches rely on matching heuristics for identifying mappings, comparing names and the natural language definitions of two concepts and checking the closeness of two concepts in the concept hierarchy [15, 13]. Based on the inter-ontology

mappings derived in mapping discovery, a merging process integrates the source ontologies and generates a concept hierarchy that embeds the knowledge encoded in the initial ones. However, as recognised in [25], deriving a meaningful concept hierarchy is a hard problem even with the ground set of inter-ontology mappings provided. Most methods that support the merging process are performed in an interactive manner with the interference of human users. In cases of large scale ontologies, the resulting class hierarchies generated tend to be inconsistent.

Another branch of research studies ontology construction with formal methods. Of particular interest here is research on FCA. FCA is a formal method for conceptual structure derivation. FCA related tools enable considerable of knowledge processing activities automated, particularly concept generation and hierarchy derivation. As a result FCA has been attracting great interest to support systematic, semi-automated development of ontologies. For example, FCA has been employed to construct domain specific ontologies in research [11, 19]. The theory of FCA has also been adapted to problems of ontology or concept hierarchy merging [25, 19, 21].

Building on advances made in [25, 21, 19], in this research we propose a FCA-based method for utility ontology construction, with the focus on treating implicit and ambiguous information. Previous work publication is either implicit on how these problems are resolved, or only addresses particular types of these problems. For example, in [25] there is an interesting discussion on attribute naming conflicts, but the authors do not address in detail how these problems are resolved.

3. FCA-based Utility Ontology Integration

With FCA theory as the backbone, we have developed a method that is designed to support utility ontology integration. The method supports automated concept hierarchy derivation and mapping identification, and it also supports ontology integration in the presence of implicit and ambiguous information. Implicit information is caused by the fact that utility companies tend to explicitly encode asset types having specific attributes, but leave others unspecified. As an example in the sewer domain, a sewer pipe is characterised by how it conveys sewerage: either by *gravity* or by *pressure*, with the gravity distribution employed more often than pressurised one. Most utility companies explicitly specify pressurised sewer pipes, but not gravity sewer pipes. If not restored in FCA, this missing information can lead to an ontology that is ill-formed, and does not correctly capture critical concepts and semantics of the domain.

Another challenge in applying FCA in ontology integration is to deal with inconsistent or ambiguous information, particularly inconsistencies/ambiguities existing with asset attributes. For example, different terms may be employed to refer to the same attribute, and attributes may be modelled at different level of detail. A simple example is that one utility data resource may model a sewer pipe as either *main* or *lateral* and another may classify it as *trunk main*, *non-truck main*, and *service*. These mismatches pose considerable difficulties in applying FCA to ontology integration.

In this section, we present a generic framework that supports ontology integration with FCA. We will describe in detail the techniques that deal with implicit and ambiguous data in Section 4 and 5. Figure 2 shows the main components of the FCA-based integration framework. The process of utility ontology development consists of three steps: Context Formation, Context Composition, and Ontology and Mapping

Derivation. Context Generation takes data and meta-data from a utility dataset as input, and generates a many valued context $K := \langle G, M, W, I \rangle$, where G contains asset types, M contains attributes of asset types, W contains attribute values, and a ternary relation $I \subseteq G \times M \times W$ contains object attribute value relationships. The next step, Information Explication, restores implicit information, which will be covered in more detail in Section 4. The final step, Conceptual Scaling, transforms many valued contexts into one valued contexts, in order for classic FCA techniques to be applicable.

Context Composition takes two contexts as input and generates an integrated concept lattice. A data dictionary that maintains utility asset and related terms is employed to disambiguate conflict attributes in context composition, which will be detailed in Section 5. The integrated context is then fed to a lattice generating component to produce a concept lattice. To prevent too many irrelevant concepts (with respect to the concepts in source contexts) from being generated, a pruned concept lattice is derived here instead of classic concept lattice of the context. A pruned concept lattice is a lattice that eliminates the concepts that have empty *intent* and *extent* [9]. The computation of the concept lattice is done with the FCA tool Galicia [24].

Ontology and Mapping Derivation takes the concept lattice generated in the previous step as input and generates an integrated ontology as well as mappings between utility asset types. Each formal concept of the lattice is a candidate for an ontology class or relation (human interaction is required in this step for decision making). The edge between two concepts indicates an *is-a* relationship. The derived ontology is represented in OWL to form a formal ontology specification. This step also generates candidate mappings between asset types. Given a formal concept, if its extent contains more than one object (asset type), then it indicates a potential mapping between these asset types.

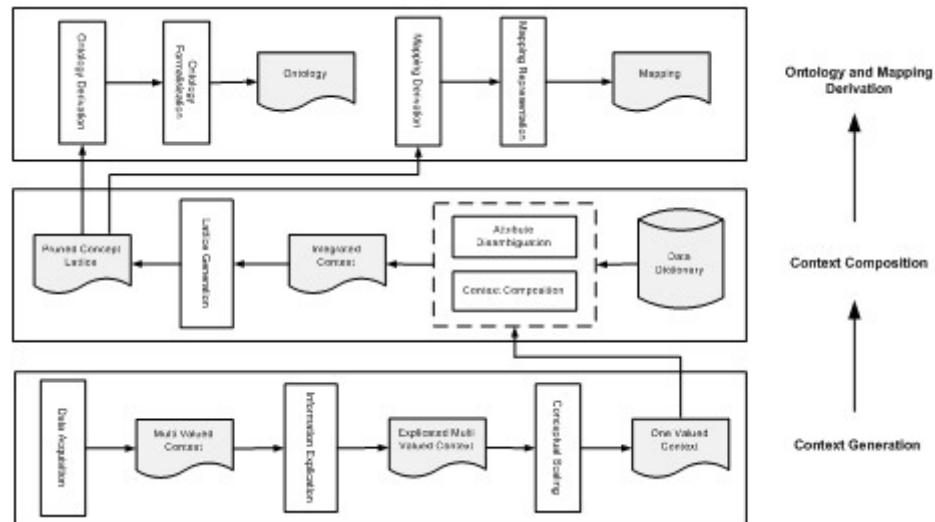


Figure 2 FCA-based Utility Ontology Integration

4. Information Explication

The data acquisition step (shown in Figure 2) acquires asset types from a utility dataset and results in a many valued context. Table 1 shows a portion of a many valued context we generated from a utility dataset. It explicitly specifies attributes such as *pressurized*, *above_ground* and *abandoned*, leaving those such as *gravity*, *underground* and *operational* implicit. Figure 3 shows the corresponding concept lattice.

Table 1

	size	what	how	position	status
sewerPipeType1	main		pressurised		
sewerPipeType2	main			above_ground	
sewerPipeType3	main	sludge			
sewerPipeType4	main				abandoned

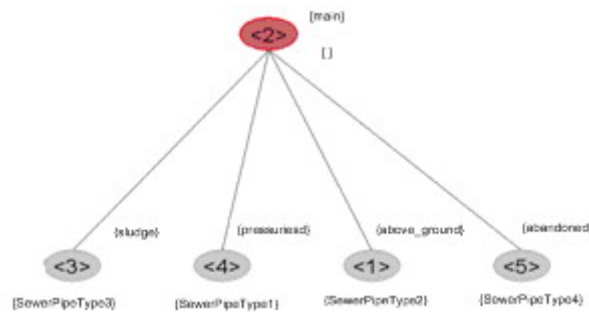


Figure 3 Concept Lattice Before Restoring Implicit Information

To restore implicit information, we use a set of domain specific rules¹, e.g., the rules to restore implicit information for the above context table are shown in Figure 4.

- 1) if a sewer is not explicitly specified as a pressurised sewer, then it is a gravity sewer;
- 2) if a sewer is not explicitly specified as an above ground sewer, then it is an underground sewer;
- 3) if a sewer is not explicitly specified as an abandoned sewer, then it is an operational sewer;
- 4) a sludge sewer is a pressurised sewer unless it is specified otherwise;

Figure 4 A Set of Rules for Restoring Implicit Information.

Several methods have been suggested in this research to apply the rules to the contexts, and each results in a concept lattice with different structure and granularity.

¹ Rules are collected based on domain knowledge and with the assistance of utility experts. The detail on the rule selections is not described here due to space limitation.

- I. This method applies all rules to each object, which results in a context with same number of objects and objects are updated with new attributes. For example, *sewerPipeType1* will have following new attribute: *wastewater*, *underground* and *operational*. Table 2 shows the updated context with this method.

Table 2

	size	what	how	position	status
sewerPipeType1	main	wastewater	Pressurised	underground	operational
sewerPipeType2	main	wastewater	Gravity	above_ground	operational
sewerPipeType3	main	sludge	Pressurised	underground	operational
sewerPipeType4	main	wastewater	Gravity	underground	abandoned

- II. This method retains original objects, and extends the context table with new objects generated by applying different combination of rules. For example, by using the first two rules shown in Figure 3, three new objects can be derived from the object *sewerPipeType1*, which is shown in Table 3. This option is potentially useful for generating a lattice with the richest semantics, but the number of objects derived is exponential in the number of rules. The number of concepts is even bigger, which grows exponentially with the numbers of objects.

Table 3

	size	what	How	Position	status
sewerPipeType1	main		Pressurised		
object1	main	wastewater	Pressurised		
object2	main		pressurised	underground	
object3	main	Wastewater	pressurised	underground	
::: :::					

- III. The third approach, which is a balance between the first and the second approaches, retains original objects, but for each object, generates a new one by applying all rules. The context generated with this approach is shown in Table 4.

Table 4

	size	what	how	Position	Status
sewerPipeType1	main		pressurised		
object1	main	wastewater	pressurised	Underground	operational
sewerPipeType2	main			above_ground	
object2	main	wastewater	gravity	above_ground	operational
sewerPipeType3	main	sludge			
object3	main	sludge	pressurised	underground	operational
sewerPipeType4	main				abandoned
object4	main	wastewater	gravity	underground	abandoned

Figure 5 shows the concept lattice which has implicit information restored with approach I. Comparing with the original concept lattice shown in Figure 3, extra formal concepts such as $(\{underground\}, \{\})$ and $(\{gravity\}, \{\})$, are identified here, which capture important semantics of the application domain and will form critical ontological concepts. We are still investigating which approach discussed above best recovers these missing semantics, which is largely performed by doing ontology coverage analysis and some evaluation with domain users.

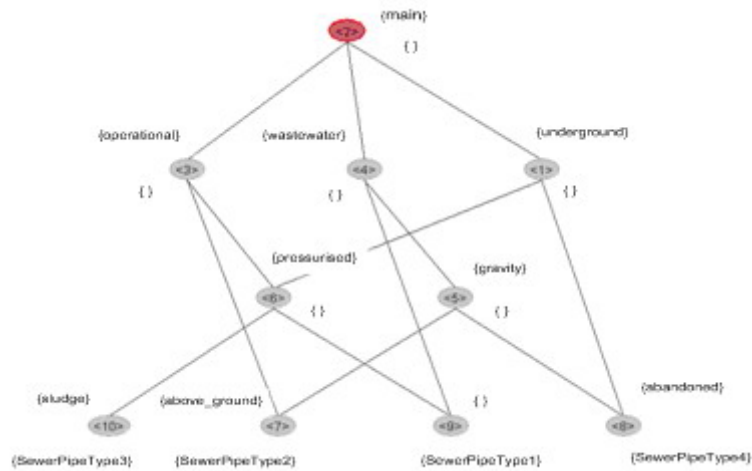


Figure 5 Concept Lattice After Restoring Implicit Information

5. Attribute Disambiguation

Context composition takes two formal contexts² as input, and generates an integrated lattice. Due to the possibility of inconsistencies arising from attributes of the source contexts, context composition is not a simple union or disjoint union operation. To deal with this, we employed a utility data dictionary. The data dictionary is developed with techniques described in [7], and it maintains a set of terms that describes utility asset types, and it also encodes terminological relationships between utility terms, such as BT/NT (Broader Term/Narrower Term), USE/UF (Use/Used For) and RT (Related Term). With the assistance of the data dictionary, we are not only able to identify whether two attributes are equivalent (by utilising USE/UF relationships), but also able to generate subsumption relationships between two attributes (by utilising BT/NT relationships). In what follows, we describe the operations that resolve attribute mismatch, and we will use the context tables shown in Figure 6 to illustrate the context composition process.

Context K ₁				Context K ₂				
	operational	abandoned	proposed recommission	live	abandoned intact	proposed	standby	
ob1	X				X			
ob2			X	X				
ob3						X		
ob4		X					X	

Figure 6 Example Source Contexts

² Which are one valued after the conceptual scaling step.

Given two contexts $K_1 := \langle G_1, M_1, I_1 \rangle$ and $K_2 := \langle G_2, M_2, I_2 \rangle$, the integrated context $K := \langle G, M, I \rangle$ is computed by first performing a disjoint union of object sets of two contexts, i.e., $G = G_1 \cup^* G_2$. For an object $(g, n) \in G$ of K , we say that (g, n) *originates* from K_1 when $n=1$ and from K_2 when $n=2$. M and I are assigned M_1 and I_1 from K_1 at this stage. That is $M \equiv M_1$ and $I \equiv I_1$. Table 5 shows the integrated context K after the above operations.

Table 5

	operational	abandoned	proposed recommission
(ob1, 1)	X		
(ob2, 1)			X
(ob3, 1)		X	
(ob1, 2)			
(ob2, 2)			
(ob3, 2)			
(ob4, 2)			

The next step expands context K with respect to attributes of K_2 . For each attribute $A_i \in M_2$ of K_2 , we perform a semantic mapping operation with attributes in K . Depending on the type of match resulted, different operations are performed:³

- I. A_i finds an equivalent attribute $A_j \in M$ of K . Such an equivalent attribute may either be specified with the same term, or with a synonym. For example for the context K_2 in Figure 5, if the attribute *live* finds an equivalent attribute *operational* in K , A_i will be unified with A_j , and context table K is expanded with existing relationships between A_i (or A_j after unification) and objects that originated from K_2 , as shown in Table 6. This context expansion is highlighted with columns and rows having a emboldened border in the table.

Table 6

	operational	abandoned	proposed recommission
(ob1, 1)	X		
(ob2, 1)			X
(ob3, 1)		X	
(ob1, 2)			
(ob2, 2)	X		
(ob3, 2)			
(ob4, 2)			

³ In many situations, complications can occur, e.g., an attribute may have multiple matches. The primitive operations described can be composed to deal with these complex cases. We will not elaborate this further here due to space limitations.

- II. A_i finds attribute A_j that is more generic to it. For example, the closest match for *abandoned intact* in K_2 is *abandoned* which is a broader term to it. In this case, the resulting context K is expanded with attribute A_i and existing relationships between A_i and objects from K_2 . New binary relationships (as shown with shaded cells) are established in K between those objects having attribute A_i from K_2 and attributes A_j (which is originally from K_1). This is shown in Table 7. The underlying theory is that if A_i is a sub term of A_j , then any object which has attribute A_i should also have attribute A_j . In our example, if a sewer pipeline is *abandoned intact*, then it is also *abandoned*.

Table 7

	operational	abandoned	proposed recommission	abandoned intact
(ob1, 1)	X			
(ob2, 1)			X	
(ob3, 1)		X		
(ob1,2)		X		X
(ob2,2)	X			
(ob3,2)				
(ob4,2)				

- III. A_i finds a matching attribute A_j that is more specific to it. For example, the closest match for the attribute *proposed* in K_2 is *proposed recommission* in K which is a narrower term for it. In this case, the context K is expanded with A_i and existing relationships between A_i and objects originating from K_2 , as shown in Table 8. New binary relationships (as shown with shaded cells) are established in K between those objects having attribute A_j (originally from K_1) and attribute A_i (which is originally from K_2).

Table 8

	operational	abandoned	proposed recommission	abandoned intact	proposed
(ob1, 1)	X				
(ob2, 1)			X		X
(ob3, 1)		X			
(ob1,2)		X		X	
(ob2,2)	X				
(ob3,2)					X
(ob4,2)					

- IV. A_i finds no match in K . For example there is no semantic match in K for the attribute *standby* in K_2 . In this case the context K is simply expanded with A_i and existing relationships between A_i and objects originating from K_2 , as shown in Table 9.

Table 9

	operational	abandoned	proposed recommission	abandoned intact	proposed	standby
(ob1, 1)	X					
(ob2, 1)			X		X	
(ob3, 1)		X				
(ob1,2)		X		X		
(ob2,2)	X					
(ob3,2)					X	
(ob4,2)						X

6. Experiments

The techniques described in the previous sections have applied to real world utility data to build an ontology for the sewer utility domain. Data that specifies sewer pipelines were collected from two utility companies. From utility dataset one, 54 utility asset types were collected; these asset types are described with 21 attributes. From utility data resource two, 49 utility asset types were collected; these asset types are described with 16 attributes. A set of rules for restoring implicit information was collected for each data resource. Our preliminary experiments demonstrated that these rules overlap considerably between two datasets, which leads to the assumption that these rules are largely domain specific rather than application specific. That is, rules collected are applicable to utility data from different utility companies and therefore can be reused in future integration. However we anticipate that due to the different encoding practices employed by utility companies, the numbers of applicable rules will differ from one dataset to another. Further experiments are still required to prove these assumptions. The approach I (discussed in Section 4) was tested for restoring implicit information, and Figure 7 shows the concept lattices for two datasets with implicit information restored (labels are eliminated here for the reasons of confidentiality and readability). Work is still ongoing in testing approaches II and III.

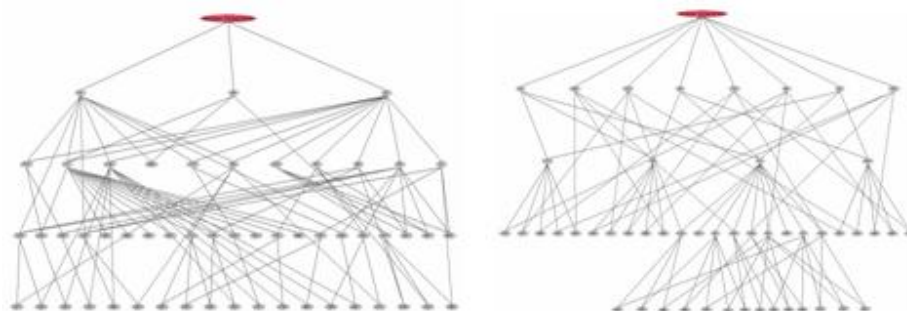


Figure 7 Concept Lattice Generated for Source Datasets

The formal contexts from the two data resources were composed to generate an integrated context with 103 objects and 26 attributes. The disambiguation results show that there were 12 Type I matches, 0 Type II matches, 2 Type III matches, and 5 type IV matches. Multi matches existed for one attribute and it had 1 equivalence match (Type I) and 2 specific matches (Type III). This revealed that utility companies tend to use common set of attributes to describe their asset types, but diversity does exist. Figure 8 shows the pruned concept lattice for the integrated context.

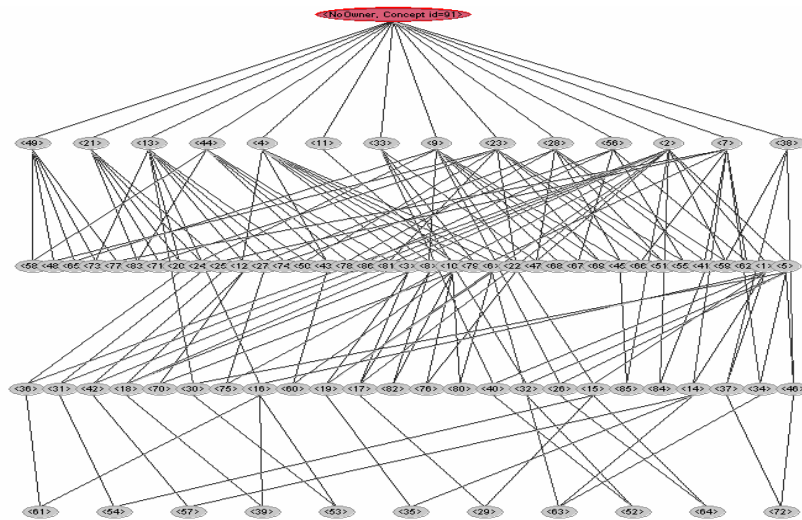


Figure 8 Integrated Concept Lattice

7. Conclusions and Future Work

This paper presents our approach to utility ontology development for the purpose of providing an improved information mapping service for underground utility apparatus records. The research employs formal concept analysis techniques as the backbone for formal concept identification and hierarchy derivation, and the method can also be used to identify utility asset type mapping cross different utilities. Techniques have been proposed to resolve implicit information and ambiguous information, and preliminary experiments have been carried out with the aim to constructing a sewer utility ontology with real world utility data. The techniques have several advantages over traditional knowledge engineering approaches for ontology construction, the key one being that it supports systematic and a semi-automated method for ontology integration.

Ongoing research is on deriving ontology concepts and the concept hierarchy from the integrated lattice with respect to the usefulness of ontology generated. Further work is required to take into account of different data modelling styles when applying proposed techniques. Other research we plan to perform in the future includes testing the techniques proposed with more utility datasets, and evaluation of the ontologies generated with the different approaches.

Acknowledgement

This work is funded by EPSRC Grant EP/C014707/1 and DTI Grant DTI/TSB 15820. Thanks are given to our utility partners for the valuable data and domain knowledge they provided.

References

- [1] A. Beck, Gaihua Fu, A. G. Cohn, B. Bennett & J. G. Stell , A framework for utility data integration in the UK, in Proceedings of *UDMS Symposium2007*, pp.261-276, 2007.
- [2] M. Burtwell, E. Faragher, D. Neville, C. Overton, C. Rogers, and T. Woodward. Locating Underground Plant and Equipment: Proposals for a Research Programme. *Technical Report, UKWIR*, 2004.
- [3] A. Doan, P. Domingos, and A.Y. Halevy, Reconciling Schemas of Disparate Data Sources: A Machine-Learning Approach. *SIGMOD record*, 30(2), pp.509-520, 2001.
- [4] Federal Geographic Data Committee, Utilities Data Content Standard. <http://www.fgdc.gov/standards/projects/FGDC-standards-Projects/utilities/utilities.pdf>.
- [5] M. Fernandez and A. Gomez-Perez and N. Juristo, METHONTOLOGY: from Ontological Art towards Ontological Engineering, in Proceedings of the AAAI97 Spring Symposium Series on Ontological Engineering, pp.33--40, 1997.
- [6] B. Ganter and R. Wille (Author), Formal Concept Analysis: Mathematical Foundations, Springer, ISBN-10: 3540627715, 1998.
- [7] G. Fu and A.G. Cohn, Semantic Integration for Mapping the Underworld, to appear in *GeoInformatics2008*, http://www.comp.leeds.ac.uk/mtu/pubs/semantic_geoinformatics08.pdf.
- [8] A. Gangemi, D. Pisanelli and G. Steve. Ontology Integration: Experiences with Medical Terminologies. in Proceedings of International Conference of FOIS, pp.163-178, 1998.
- [9] R. Godin, H. Mili , G. Mineau, R. Missaoui, A. Arfi and T. Chau, Design of class hierarchies based on concept (Galois) lattices, Theory and Practice of Object Systems, Vol.4(2), pp.117 – 134, 2006.
- [10] D.M. Jones, T.J.M. Bench-Capon and P.R.S. Visser, Methodologies for Ontology Development, in Proc. IT&KNOWS Conference, XV IFIP World Computer Congress, 1998.
- [11] N. Jyotirmaya, S. Timothy, S. Kumara And S. Shooter, A methodology for product family ontology development using formal concept analysis and web ontology language, in Journal of computing and information science in engineering, vol.6(2), pp.103-113, 2006.
- [12] Y. Kalfoglou and M. Schorlemmer, Ontology Mapping: The State of the Art, in *Proceedings of Dagstuhl Seminar -Semantic Interoperability and Integration*, 2005.
- [13] M. Kavouras and M Kokla, A method for the formalization and integration of geographical categorizations, in *International Journal of GIS*, 16(5), pp.439 – 453, 2002.
- [14] M Lenzerini. Data Integration: A Theoretical Perspective. In *Proc. PODS*, pp 233—246, 2002.
- [15] J. Madhavan, P. Bernstein and E. Rahm., Generic schema matching with cupid, in *The VLDB Journal*, pp.49-58, 2001.
- [16] N. Noy and D. McGuinness. Ontology Development 101: A Guide to Creating Your First Ontology, Technical Report, Stanford, 2001.

- [17] N. Noy and M. Musen, PROMPT: Algorithm and Tool for Automated Ontology Merging and Alignment, in Proceedings of the 16th National Conference on Artificial Intelligence, pp.450-455, 2000.
- [18] N. Noy, Semantic Integration: a Survey of Ontology-Based Approaches, in *SIGMOD Record*,33(4), pp.65-70, 2004.
- [19] I. Schmitt and G. Saake, Merging Inheritance Hierarchies for Database Integration, in Proceedings of the 3rd IFCIS International Conference on Cooperative Information Systems, pp.322 - 331, 1998.
- [20] H. Pinto and J. Martins, A methodology for ontology integration. in Proceedings of the 1st international conference on Knowledge capture, pp.131-138, 2001.
- [21] G Stumme and A Maedche, FCA-MERGE: Bottom-Up Merging of Ontologies, in Proc. International Joint Conference On AI, pp.225-234, 2001.
- [22] M. Uschold and M. Grüninger, Ontologies: principles, methods, and applications, in Knowledge Engineering Review, Vol 11(2), pp.93-155, 1996.
- [23] M. Uschold, Towards a Methodology for Building Ontologies, in Workshop on Basic Ontological Issues in Knowledge Sharing, 1995.
- [24] P. Valtchev, D. Grosser, C. Rouse and M. Hacene, Galicia: An Open Platform for Lattices, <http://citeseer.ist.psu.edu/669715.html>.
- [25] M. Rouane, P. Valtchev, H. Sahraoui and M. Huchard, Merging conceptual hierarchies using concept lattices, MASPEGHI Workshop, 2004.
- [26] R. Wille, Restructuring lattice theory: an approach based on hierarchies of concepts, in *Ordered Sets*. Reidel, Dordrecht-Boston, pp.445-470, 1982.