Developing Non-monotonicity in Formal Concept Analysis

Lucas Carr
Dept. of Computer Science
University of Cape Town
crrluc003@myuct.ac.za

1 INTRODUCTION

Formal concept analysis (FCA) provides a framework, grounded in lattice theory, for mathematically reasoning about *formal concepts* and their hierarchies [3, 4, 8]. The matter of *concepts* has largely been a Philosophical concern: the notion of a concept as the dualism between *intension* and *extension* has foundations in Aristotle's *Organon* and, much later on, in the *Logic of Port-Royal* [1, 8]. In this view, the extension of a concept contains to those "things" which one might refer to as instances of the concept. Dually, intension describes the meaning, or sense, of the concept.

Formal concept analysis, which adopts this view of concepts, introduces a *formal context*: a triple consisting of a finite set of objects G, attributes M, and a binary relation, $I \subseteq G \times M$, which indicates that a particular object has a respective property [3, 4]. A *formal concept* is a pair of sets - the formal concept extension and intension, respectively. The set of all concepts, when ordered by the *sub/super-concept* relation, form the *concept lattice* used for analysis.

Another important topic in FCA is the discovery of *implications* pertaining to–or, *respected* by–a context [3, 8]. Implications are used to express correspondencies that exist between (sets) of attributes in a given context. The notion of a context *respecting* an attribute implication is analagous to that of *entailment* in classical logic [4]. As such, *respecting* describes a monotonic notion of consequence.

Discussion and work on non-monotonic propositional, first-order, and description logic is a well established topic in artificial intelligence, [2, 5–7, 9]. However, there does not appear to be any effort to introduce this expressivity to the attribute logic of FCA. In part, this may be due to FCA lying somewhere inbetween fields - not quite reasoning, and not solely analysis either. That is not to say that increasing the expressivity of FCA's attribute logic would not be beneficial: association rules are a fairly blunt way of gaining some non-monotonicty in FCA. However, they acquire meaning through majority rule - where we would like something more precise. Applications of FCA: text mining, web mining, and ontology engineering

2 BACKGROUND

2.1 Formal Concept Analysis

The starting point of FCA is the *formal context*. A formal context is a tripple consisting of a finite set of objects, a finite set of attributes, and a relation which describes when an object g 'has' an attribute m.

Definition 2.1. A *formal context* is a tripple, denoted $\mathbb{K} = (G, M, I)$, where G is a finite set of objects, M is a finite set of attributes, and $I \subseteq G \times M$ is an incidence relation on the cartesian product of G and M.

For contexts of reasonable size, a cross-table is frequently used as a visual aid. Each row represents an object, and each column an attribute. A '×' is used to indicate where an object has an attribute [3, 4]. See Table 1 for an example. For any subset of objects (attributes) we use a derivation operator to describe the attributes (objects) common to all memberss of that subset. These derivation operators form a antitone Galois connection between the powersets $\mathcal{P}(G)$ and $\mathcal{P}(M)$ [3].

Definition 2.2. In a context, $\mathbb{K} = (G, M, I)$, we define two *derivation operators*, both denoted by $(\cdot)'$, as follows:

$$A \subseteq G: \quad A' := \{m \in M \mid \forall g \in A : (g, m) \in I\}$$
$$B \subseteq M: \quad B' := \{g \in G \mid \forall m \in B : (g, m) \in I\}$$

In case of a singleton set, $\{g\}$, the braces are omitted. If g is an object, then g' describes the *object intent*; if g is an attribute, g' is the *attribute extent*. A derivation operator can be applied to the result of an earlier derivation operator, this is then referred to as a *double-prime operator*, and expressed as $(\cdot)''$. Any such set is closed, and so each derivation operator describes a closure system on G and M, respectively [3, 4, 8]. As such, the double-prime operator $(\cdot)''$ satisfies:

$$\text{Monotonocity: if } A_1 \subseteq A_2 \text{ then } A_1'' \subseteq A_2'' \tag{1}$$
$$\text{Extensivity: } A \subseteq A'' \tag{2}$$
$$\text{Idempotency: } A'' = (A'')'' \tag{3}$$

Definition 2.3. A *formal concept* of a context, $\mathbb{K} = (G, M, I)$, is a pair (A, B) where $A \subseteq G$ and $B \subseteq M$ for which $A' = B$ and $B' = A$. Then A is the *concept extent* and B is the *concept intent*. The set of all concepts of a context is given by $\mathfrak{B}(G, M, I)$.

There is obvious redundancy in this definition of concepts: if (A, B) is a formal concept, it could equivalently be given by (A, A') or (B', B). Moreover, for an arbitrary set A of objects (attributes), A' defines a concept intent (extent), and A'' defines a concept extent (intent) [4].

Definition 2.4. Let (G, M, I) be a formal context, then for every object, $g \in G$, we define a *object concept* as

$$\gamma g := (g'', g')$$

and for each attribute, $m \in M$, we define the *attribute concept* as

$$\mu m := (m', m'')$$

The idea of sub and super-concepts gives rise to a natural partial order on $\mathfrak{B}(G, M, I)$. Specifically, if (A_1, B_1) and (A_2, B_2) are concepts in $\mathfrak{B}(G, M, I)$, then (A_1, B_1) is a *sub-concept* of (A_2, B_2) iff $A_1 \subseteq A_2$ (equivalently, $B_2 \subseteq B_1$). Obviously, (A_2, B_2) is respectively a *super-concept*, and we say that $(A_1, B_1) \leq (A_2, B_2)$ [3]. Then,

$\mathfrak{B}(G, M, I)$ and the relation $\leq$ form a complete lattice called the *concept lattice.* [link to concept lattice in appendix]

The concept lattice can be used to read-off correspondencies between sets of attributes - we call these relationships *attribute implications*, which use a fragment of *attribute logic* and can reasonably be thought of as material implications.

Definition 2.5. Let M be an arbitrary set. An *implication* over M takes the form $B_1 \to B_2$ where $B_1, B_2 \subseteq M$. Another set $C \subseteq M$, *respects* the implication if $B_1 \nsubseteq C$ or $B_2 \subseteq C$. Then, $C \models B_1 \to B_2$.

Definition 2.6. For $\mathbb{K} = (G, M, I)$ and an implication i over M, the implication is *valid* in the formal context iff for every object $g \in G$ it is the case that the object intent, g', respects the implication. Then, $\mathbb{K} \models i$. If $i := B_1 \to B_2$, then the following are equivalent

$$\begin{aligned} \mathbb{K} &\models B_1 \to B_2 \\ B_1' &\subseteq B_2' \\ B_2 &\subseteq B_1'' \end{aligned}$$

2.2 Classical Notions of Consequence

In a classical setting, *logical consequence* describes when knowing one formula, α, is sufficient to conclude another, β - then β is a logical consequence of α.

In classical logic, the notion of *logical consequence* describes when one formula allows one to conclude another. If $\alpha, \beta \in \mathcal{L}$ are two formulae in the language, then $\alpha \models \beta$ iff for every valuation $u \in \mathcal{U}$ if $u \Vdash \alpha$ then $u \Vdash \beta$, equivalently that the models of α are a subset of the models of β. We can similarly define a notion of *classical entailment* which describes when a formula is a logical consequence of set of formulae, or a knowledge-base.

Definition 2.7. We say that for two formulae, α, β, in the language $\mathcal{L}$, β is a *logical consequence* of α, written $\alpha \models \beta$ iff for every valuation $u \in \mathcal{U}$ where $u \Vdash \alpha$ then also $u \Vdash \beta$.

Definition 2.8. Given a knowledge-base $\mathcal{K}$ and a formula α, the knowledge-base entails α, written $\mathcal{K} \models \alpha$, iff every for every valuation $u \in \mathcal{U}$ where $u \Vdash \mathcal{K}$ it is also the case that $u \Vdash \alpha$.

We introduce a consequence operator, $\mathcal{C}n$, where $\mathcal{C}n(\mathcal{K})$ describes a closed-set containing everything that can be entailed from $\mathcal{K}$. With this in mind, any notion of consequence which satisfies the condtions below is referred to as a *Tarskian operation.*

(1) Monotonocity: *if* $\mathcal{K} \subseteq \mathcal{K}'$ *then* $\mathcal{C}n(\mathcal{K}) \subseteq \mathcal{C}n(\mathcal{K}')$
(2) Idempotence: $\mathcal{C}n(\mathcal{K}) = \mathcal{C}n(\mathcal{C}n(\mathcal{K}))$
(3) Inclusion: $\mathcal{K} \subseteq \mathcal{C}n(\mathcal{K})$

Consequence relations are a more removed way of describing some notion of logical consequence. These relations are a (possibly infinite) set of ordered pairs containing formulae in the language, where the first element is the antecedent and the second element is the consequent. A consequence relation might look like $\{(\alpha_0, \beta_0), \ldots, (\alpha_n, \beta_n) \ldots\}$. The relation does not give a semantic or syntactic notion of consequence. Rather, a consequence relation satisfies certain properties which describe some pattern of reasoning - as such, a consequence relation might be a characterisation of some notion of consequence. Obviously, not all sets of pairs would define a meaningful pattern of reasoning. At a minimum, a meaningful consequence relation, Γ, should satisfy

(1) Reflexivity: $(\alpha, \alpha) \in \Gamma$
(2) Cut: *if* $(\alpha, \beta \wedge \gamma) \in \Gamma$ *and* $(\gamma \wedge \delta, \mu) \in \Gamma$ *then* $(\alpha \wedge \delta, \beta \wedge \mu) \in \Gamma$

2.3 Non-monotonic Consequence

3 MOTIVAION AND OBJECTIVES

Algorithm 1 Computing the tolerance partition on objects G

Require: A formal context, $\mathbb{K} = (G, M, I)$;
Require: A set of defeasible implications, Δ;
Ensure: A tolerance partition $(R_0, \ldots, R_n)$ on G;
1: $P_0 \leftarrow G$;
2: $i \leftarrow 0$;
3: $\Delta \leftarrow \text{Material}(\Delta)$;
4: **while** $P_{i-1} \neq P_i$ **do**
5: $\quad P_{i+1} \leftarrow \{g \in P_i \mid \exists \delta \in \Delta g' \text{ s.t. } g' \not\models \delta\}$;
6: $\quad R_i \leftarrow P_i \setminus P_{i+1}$;
7: $\quad \Delta_i \leftarrow \{(\alpha \to \beta) \in \Delta \mid \exists g \in P_i \text{ s.t. } \alpha \subseteq g'\}$;
8: $\quad \Delta \leftarrow \Delta \setminus \Delta_i$;
9: **end while**
10: **if** $P_{i-1} = \varnothing$ **then**
11: $\quad n \leftarrow i - 1$;
12: **else**
13: $\quad n \leftarrow i$;
14: **end if**
15: **return** $(R_0, \ldots, R_n)$;

To clarify some points on

- Line 3: this step turns Δ into a set of material implications;
- Line 5: this step puts into ranking $i + 1$ every object which **does not** respect Δ;
- Line 7: this sets Δ_i to be the set of formulae in Δ which have their antecedent satisfied by an object in P_i (implicitly all objects in P_i satisfy all formulae in Δ, but we only care about the ones which have their antecedent satisfied);
- We update Δ to be those formulae which are not yet "dealt with".

REFERENCES

[1] Charles Castonguay. 2012. *Meaning and existence in mathematics*. Vol. 9. Springer Science & Business Media.

[2] Kenneth G Ferguson. 2003. Monotonicity in practical reasoning. *Argumentation* 17 (2003), 335–346.

[3] B Ganter. 1999. Formal Concept Analysis: Mathematical Foundations. (1999).

[4] Bernhard Ganter, Sergei Obiedkov, Sebastian Rudolph, and Gerd Stumme. 2016. *Conceptual exploration*. Springer.

[5] Laura Giordano, Valentina Gliozzi, Nicola Olivetti, and Gian Luca Pozzato. 2015. Semantic characterization of rational closure: From propositional logic to description logics. *Artificial Intelligence* 226 (2015), 1–33.

[6] Sarit Kraus, Daniel Lehmann, and Menachem Magidor. 1990. Nonmonotonic reasoning, preferential models and cumulative logics. *Artificial intelligence* 44, 1-2 (1990), 167–207.

[7] Daniel Lehmann and Menachem Magidor. 1994. What does a conditional knowledge base entail? *Artificial Intelligence* 68, 2 (1994).

[8] Sebastian Rudolph. 2007. *Relational exploration. Combining description logics and formal concept analysis for knowledge specification.*

[9] Yoav Shoham. 1987. Nonmonotonic Logics: Meaning and Utility.. In *IJCAI*, Vol. 10. Citeseer, 388–393.

APPENDIX

	Aves	Mammal	Reptile	Aerial	Aquatic	Terrestrial	Migratory	Solitary	Carnivore	Eggs
Bat		×		×					×	
Crocodile			×		×	×			×	×
Crow	×			×						×
Hippo		×			×	×		×		
Ostrich	×					×		×		×
Penguin	×				×	×	×		×	×
Platypus		×			×	×		×	×	×
Snake			×			×		×	×	×
Swallow	×			×			×	×	×	×
Whale		×			×		×	×	×	

Table 1: A formal context describing animals (objects) and some of their features (attributes)

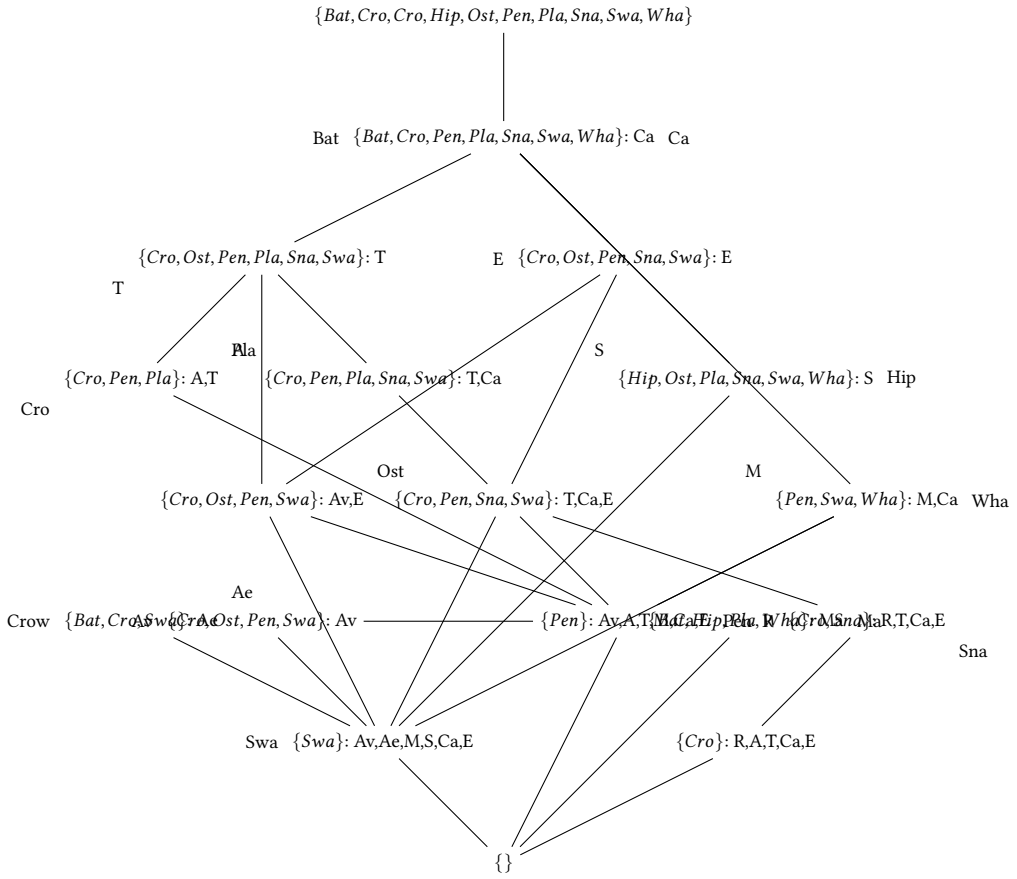

Figure 1: Concept Lattice for Animal Attributes