On logical and extensional characterizations of attributed feature models

On Logical and Extensional Characterizations of Attributed Feature Models

Ferruccio Damiani[a], Michael Lienhardt[b], Luca Paolini[a,*]

[a] *University of Turin, Turin, Italy*
[b] *ONERA, Palaiseau, France*

Abstract

Software-intensive systems can have thousands of interdependent configuration options across different subsystems. Feature models (FMs) allow designers to organize the configuration space by describing configuration options using interdependent features: a feature is a name representing some functionality and each software variant is identified by a set of features. Attributed feature models (AFMs) extend FMs to describe the, possibly constrained, choice of a value from domains such as integers or strings: each attribute is associated to one feature, and when the feature is selected then the attribute brings some additional information relative to the selected features. Different representations of FMs and AFMs have been proposed in the literature. In this paper we focus on the logical representation (which works well in practice) and the extensional representation (which has been recently shown well suited for theoretical investigations). We provide an algebraic and a logical characterization of operations and relations on FMs and AFMs, and we formalize the connection between the two characterizations as monomorphisms from lattices of logical FMs and AFMs to lattices of extensional FMs and AFMs, respectively. This formalization sheds new light on the correspondence between the algebraic and logical characterizations of operations and relations for FMs and AFMs. It aims to foster the development of a formal framework for supporting practical exploitation of future theoretical developments on FMs, AFMs and multi software product lines.

Keywords: Feature model, Attributed feature model, Boolean lattice, Composition, Configurable software, Logic, Software product line

1. Introduction

Software-intensive systems can have thousands of interdependent configuration options across different subsystems. In the resulting configuration space,

*Corresponding author
Email addresses: `ferruccio.damiani@unito.it` (Ferruccio Damiani), `michael.lienhardt@onera.fr` (Michael Lienhardt), `luca.paolini@unito.it` (Luca Paolini)

different software variants can be obtained by selecting among these configuration options and accordingly assembling the underlying subsystems. The interdependencies between options are dictated by corresponding interdependencies between the underlying subsystems [1].

Feature models (FMs) [2] allow developers to organize the configuration space and facilitate the construction of software variants by describing configuration options using interdependent *features* [3]: a feature is a name representing some functionality, a set of features is called a *configuration*, and each configuration that fulfills the interdependencies expressed by the FM, called a *product*, identifies a software variant.

Attributed feature models (AFMs) [4] extend FMs to describe the, possibly constrained, choice of a value from domains such as integers or strings. Each attribute is associated to one feature, and if the feature is selected then the attribute brings some additional information relative to the product (identified by selected features). For instance, an "SSD memory" feature may have a "capacity" attribute that should be set to a value greater than 512 GB whenever the feature "pro" is selected.

Software-intensive systems can comprise thousands of features and several subsystems [5, 6, 7, 8]. The design, development and maintenance of FMs with thousands of features can be simplified by representing large FMs as sets of smaller interdependent FMs [6, 9] that we call *fragments*. To this aim, several representations of FMs have been proposed in the literature (see, e.g., Batory [2] and Sect. 2.3 of Apel *et al.* [1]) and many approaches for composing FMs from fragments have been investigated [10, 11, 12, 13, 14].

In this paper we focus on the logical representation (which works well in practice [15, 16, 17]) and the extensional representation (which has been recently shown well suited for theoretical investigations [18, 19]).

The starting point of this investigation is a novel partial order between FMs, that we call the FM *fragment relation*. It is induced by a notion of FM composition that has been used to model industrial-size configuration spaces [18, 19], such as the configuration space of the Gentoo source-based Linux distribution [20], that consists of many configurable packages (the March 1st 2019 version of the Gentoo distribution comprises 671617 features spread across 36197 FMs). We exploit this partial order to provide an algebraic characterization of operations and relations on FMs and AFMs. Then, we provide a logical characterizations of them and formalize the connection between the two characterizations as monomorphisms from lattices of logical FMs and AFMs to lattices of extensional FMs and AFMs, respectively. This paper is an extended version of prior work [21] which does not consider AFMs.

The remainder of this paper is organized as follows. In Section 2 we recollect the necessary background and introduce the fragment relation for FMs and AFMs. In Section 3 we present the algebraic characterization of operations and relations on FMs and AFMs. In Section 4 we present the logical characterization of the operations and relations on FMs together with a formal account of the connection with the algebraic characterization, and in Section 5 we extend these results to AFMs. We discuss related work in Section 6, and conclude the paper

in Section 7 by outlining planned future work.

The material about FMs and AFMs is presented in distinct sections, all the sections about FMs or AFMs contain "FM" or "AFM" in the title, and the sections about FMs do not depend on the sections about AFMs. So, a reader may decide to first consider the material on FMs and then the extension to AFMs.

2. Basic Notions

We first recall the logical and the extensional representations of FMs (in Section 2.1) and AFMs (in Section 2.2) together with the composition operation for FMs (in Section 2.3) and AFMs in Section 2.4). Moreover, Section 2.3 and Section 2.4 include the formalization of a novel partial order relation on FMs and AFMs, respectively.

2.1. FM Representations

In this section, we focus on the logical and on the extensional representations of FMs (see, e.g., Batory [2] and Sect. 2.3 of Apel *et al.* [1] for a discussion about other representations).

Definition 2.1 (FM, logical representation). A *logical FM* Φ is a pair $(\mathcal{F}, \phi)$ where $\mathcal{F}$ is a set of features and ϕ is a propositional formula whose variables x are elements of $\mathcal{F}$:

$$\phi ::= x \mid \phi \wedge \phi \mid \phi \vee \phi \mid \phi \rightarrow \phi \mid \neg\phi \mid \mathit{false} \mid \mathit{true}\,.$$

We call *products* of Φ the sets of features $p \subseteq \mathcal{F}$ such that ϕ is satisfied by p, i.e., ϕ is satisfied by assigning value *true* to the variables x in p and *false* to the variables in $\mathcal{F} \setminus p$.

Example 2.2 (A logical representation of the glibc FM). Gentoo packages can be configured by selecting features (called *use flags* in Gentoo), which may trigger dependencies or conflicts between packages. Current versions of the *glibc* library, that contains the core functionalities of most Linux systems, are provided by the package sys-libs/glibc (abbreviated to glibc in the sequel). This package has many dependencies, including (as expressed in Gentoo's notation):

doc? (sys−apps/texinfo) vanilla?(!sys−libs/timezone−data)

This dependency expresses that glibc requires the texinfo documentation generator (provided by any version of the sys-apps/texinfo package) whenever the feature doc is selected and if the feature vanilla is selected, then glibc conflicts with any version of the time zone database (as stated with the !sys-libs/timezone-data constraint). These *dependencies* can be expressed by a FM $(\mathcal{F}_{\mathsf{glibc}}, \phi_{\mathsf{glibc}})$ where

$$\mathcal{F}_{\mathsf{glibc}} = \{\mathsf{glibc}, \mathsf{texinfo}, \mathsf{tzdata}, \mathsf{glibc{:}doc}, \mathsf{glibc{:}v}\}$$
$$\phi_{\mathsf{glibc}} = \mathsf{glibc} \wedge (\mathsf{glibc{:}doc} \rightarrow \mathsf{texinfo}) \wedge (\mathsf{glibc{:}v} \rightarrow (\neg\mathsf{tzdata}))$$

Here, the feature glibc represents the glibc package; texinfo represents any sys-apps/texinfo package; tzdata represents any version of the sys-libs/timezone-data package; and glibc:doc (resp. glibc:v) represents the glibc's doc (resp. vanilla) use flag.

The logical representation of FMs works well in practice [15, 16, 17]. Recently, Schröter *et al.* [18] pointed out that using an extensional representation of FMs simplifies the presentation of FM concepts.

Definition 2.3 (FM, extensional representation). An *extensional FM* $\mathbb{F}$ is a pair $(\mathcal{F}, \mathcal{P})$ where $\mathcal{F}$ is a set of features and $\mathcal{P} \subseteq 2^{\mathcal{F}}$ a set of products.

Example 2.4 (An extensional representation of the glibc FM). The FM of Example 2.2 can be given an extensional representation $\mathbb{F}_{\mathsf{glibc}} = (\mathcal{F}_{\mathsf{glibc}}, \mathcal{P}_{\mathsf{glibc}})$ where $\mathcal{F}_{\mathsf{glibc}}$ is the same as in Example 2.2 and

$$\begin{aligned}\mathcal{P}_{\mathsf{glibc}} = &\{\{\mathsf{glibc}\}, \{\mathsf{glibc, texinfo}\}, \{\mathsf{glibc, tzdata}\}, \{\mathsf{glibc, texinfo, tzdata}\}\} \cup \\ &\{\{\mathsf{glibc, glibc{:}doc, texinfo}\}, \{\mathsf{glibc, glibc{:}doc, texinfo, tz\text{-}data}\}\} \cup \\ &\{\{\mathsf{glibc, glibc{:}v}\}, \{\mathsf{glibc, glibc{:}v, texinfo}\}, \{\mathsf{glibc, glibc{:}doc, glibc{:}v, texinfo}\}\}.\end{aligned}$$

In the description of $\mathcal{P}_{\mathsf{glibc}}$, the first line contains products with glibc but none of its use flags are selected, so texinfo and tz-data can be freely installed; the second line contains products with the use flag doc selected in glibc, so a package of sys-apps/texinfo is always required; the third line contains products with the use flag vanilla selected in glibc, so no package of sys-libs/timezone-data is allowed; the last product in third line includes both glibc's use flags selected, so sys-apps/texinfo is mandatory and sys-libs/timezone-data forbidden.

The following definition introduces the extensional representation of the empty FM and of two distinguished kinds of FMs.

Definition 2.5 (Empty, void and trivial FMs). The *empty* FM is $\mathbb{F}_{\emptyset} = (\emptyset, \{\emptyset\})$ — it has no features and has just the empty product $\emptyset$. A FM is *void* if it has no products, i.e., it is of the form $\mathbf{void}(\mathcal{F}) = (\mathcal{F}, \emptyset)$, for some set of features $\mathcal{F}$. A FM is *trivial* if it has all the possible products, i.e., it is of the form $\mathbf{trivial}(\mathcal{F}) = (\mathcal{F}, 2^{\mathcal{F}})$, for some set of features $\mathcal{F}$ (it is worth observing that the empty FM is trivial).

Note that the logical representations of $\mathbf{void}(\mathcal{F})$ and $\mathbf{trivial}(\mathcal{F})$ are $(\mathcal{F}, \mathsf{false})$ and $(\mathcal{F}, \mathsf{true})$, respectively.

2.2. AFM Representations

An AFM is an FM extended by adding to features some attributes. A product of an AFM, called an *attributed-product*, is a product where each feature is augmented by a suitable value for each of its attributes.

In order to improve readability, we start by introducing the notion of AFM extensional representation which is simpler than that the logical one.

Definition 2.6 (AFM, extensional representation). An *extensional AFM* $\mathbb{A}$ is a 6-tuple $(\mathcal{F}, \mathcal{A}, \alpha, \mathcal{D}, \delta, \mathcal{V})$ where $\mathcal{F}$ is a set of features, $\mathcal{A}$ is the set of attributes, $\alpha : \mathcal{A} \to \mathcal{F}$ is a function mapping each attribute to a feature, $\mathcal{D}$ is the set of domains on which attributes can range over, $\delta : \mathcal{A} \to \mathcal{D}$ is a function associating each attribute to a domain, and the set of the attributed-products $\mathcal{V}$ contains pairs (p, v) such that $p \subseteq \mathcal{F}$ and $v : \alpha^{-1}(p) \to \cup\mathcal{D}$ such that $v \propto \delta$ (i.e. v is a function associating to each attribute a in $\alpha^{-1}(p)$ a value in $\delta(a)$). We write $\mathcal{P}_{\mathcal{V}}$ for the set $\{p \mid (p, v) \in \mathcal{V}\}$ of the attribute-free products of $\mathcal{V}$.

Example 2.7 (An extensional representation of the glibc AFM). Gentoo packages have two main attribute categories: a *version* local to each package, and a *keyword* which states on which hardware that package can be installed.

To illustrate these attributes, let us integrate them in our extensional presentation of the *glibc* FM, given in Example 2.4: we first add a new attribute glibc_v for the version of the *glibc* and link it to the glibc feature; and for the keyword, we add a global mandatory feature hw to model the hardware configuration and link the $\mathsf{keyword}$ attribute to it. Currently, the set of possible versions for *glibc* are: $d_{\mathsf{glibc}} = \{$2.32-r6, 2.32-r7, 2.32-r8, 2.33, 9999$\}$. And the possible architectures supported by Gentoo are: $d_{\mathsf{hw}} = \{$amd64, x86, alpha, arm, arm64, hppa, ia64, ppc, ppc64, sparc$\}$. Thus, $\mathbb{A}_{\mathsf{glibc}} = (\mathcal{F}'_{\mathsf{glibc}}, \mathcal{A}_{\mathsf{glibc}}, \alpha_{\mathsf{glibc}}, \mathcal{D}_{\mathsf{glibc}}, \delta_{\mathsf{glibc}}, \mathcal{V}_{\mathsf{glibc}})$ is the AFM for *glibc*, where:

$$\mathcal{F}'_{\mathsf{glibc}} = \mathcal{F}_{\mathsf{glibc}} \cup \{\mathsf{hw}\}, \qquad \mathcal{A}_{\mathsf{glibc}} = \{\mathsf{glibc}_v, \mathsf{keyword}\}, \qquad \mathcal{D}_{\mathsf{glibc}} = \{d_{\mathsf{glibc}}, d_{\mathsf{hw}}\},$$
$$\alpha_{\mathsf{glibc}} = [\mathsf{glibc}_v \mapsto \mathsf{glibc}, \mathsf{keyword} \mapsto \mathsf{hw}], \quad \delta_{\mathsf{glibc}} = [\mathsf{glibc}_v \mapsto d_{\mathsf{glibc}}, \mathsf{keyword} \mapsto d_{\mathsf{hw}}],$$
$$\mathcal{V}_{\mathsf{glibc}} = \{(p \cup \{\mathsf{hw}\}, [\mathsf{keyword} \mapsto k, \mathsf{glibc}_v \mapsto v]) \mid p \in \mathcal{P}_{\mathsf{glibc}},\ k \in d_{\mathsf{hw}}, v \in d_{\mathsf{glibc}}\}$$

with $\mathcal{F}_{\mathsf{glibc}}$ and $\mathcal{P}_{\mathsf{glibc}}$ as defined in Example 2.4.

The logical representation of AFM considers constraints over features and attributes. In the literature [22, 23, 12, 24] such constraints have been expressed in the language used for *Constraint Satisfaction Problem* (CSP) [25, 26]. A CSP can be defined as a triple $(\mathcal{A}, \mathcal{D}, C)$ where $\mathcal{A}$ is a finite set of variables, $\mathcal{D}$ is a finite set of domains (one for each variable) and C is a set of constraints defined on $\mathcal{A}$. A solution to a CSP is an assignment of a value to every variable, which is domains and constraints respecting. This language is a quantifier-free first-order logic that can be formulated in a programming flavour [27, 23, 28].

To simplify the presentation (and without loss of generality), in this paper we only consider domains involving integers. Namely, we consider:

- *attribute-expression*, namely $\mathsf{e} ::= n \mid a \mid \mathsf{e}\ op\ \mathsf{e}$ where constants n are values in our domains, variables a are attributes, and operations op range over $\{+, -, *, \mathtt{div}, \mathtt{mod}\}$; and
- *attribute-constraints* (i.e., constraints involving features and attributes) extending the propositional formulas of Definition 2.1 with predicative-propositions $\mathsf{e}_1\ rel\ \mathsf{e}_2$, where *rel* is a predicate in $\{=, \neq, <, \leq, >, \geq\}$.

As pointed out in the literature [12, 27, 24, 29, 30], in order to avoid semantic issues "each attribute is treated like a variable that is always defined, even if the feature that declares it is not part of the product" [12, p.1138]. Accordingly, for each domain D in $\mathcal{D}$, we assume a default value $\mathtt{d}(D)$ that is used as value of the attributes of the deselected features when checking whether an attributed-constraint is satisfied by a given attributed-product. It is natural to expect that constraints are formulated to be invariant w.r.t. the choice of the default values.

In the examples, to improve readability, we name each attribute prefixing the name of the feature to which is associated.

Example 2.8 (Constraints over features and attributes). Let f_1, f_2 be features, let $f_1.a, f_2.b$ be attributes of domain $\mathbb{N}_{10}$ (the set of the natural numbers from 0 to 9) with $\mathtt{d}(\mathbb{N}_{10}) = 0$. Then, the attribute-constraint $f_1 \wedge \neg f_2 \wedge ((f_1.a \geq (f_2.b + 1)) \vee \neg(f_1.a < 5))$ describes the 9 attribute products $(f_1, \{a \mapsto i\})$ for $1 \leq i \leq 9$. In fact, when f_2 is deselected, the default value 0 is used for $f_2.b$.

In order to support the formulation of some operations on AFMs, we consider existentially quantified attributed-constraints of the form $\exists a{:}D.\psi$, where D is a domain in $\mathcal{D}$. It is worth observing that, whenever D finite, then $\exists a{:}D.\psi$ is logically equivalent to $\psi[a := n_1] \vee \cdots \vee \psi[a := n_k]$, where the n_i $(1 \leq i \leq k)$ are the elements of D. Therefore, when all the domains in $\mathcal{D}$ are finite, the existential quantification construct can be considered as syntactic sugar.

Accordingly, we formalize the logical representation of AFMs as follows.

Definition 2.9 (AFM, logical representation). A *logical AFM* Ψ is a 7-tuple $(\mathcal{F}, \mathcal{A}, \alpha, \mathcal{D}, \delta, \mathtt{d}, \psi)$ where $\mathcal{F}$ is a set of features, $\mathcal{A}$ is the set of attributes, $\alpha : \mathcal{A} \to \mathcal{F}$ is a function mapping each attribute to a feature, $\mathcal{D}$ is the set of (non-empty) domains on which attributes can range over, $\delta : \mathcal{A} \to \mathcal{D}$ is a function associating to each attribute a domain, $\mathtt{d} : \mathcal{D} \to \cup\mathcal{D}$ is a function associating to each domain D one of its values (the default value) such that $\mathtt{d} \propto \delta$ (viz. $\mathtt{d}$ is a function associating to each domain D a value in D), and ψ is an attribute-constraint formula generated from:

$$\psi \ ::= \ x \ \mid \ \mathsf{e}\ rel\ \mathsf{e} \ \mid \ \exists a{:}D.\psi \ \mid \ \psi \wedge \psi \ \mid \ \psi \vee \psi \ \mid \ \psi \to \psi \ \mid \ \neg\psi \ \mid \ \mathit{false} \ \mid \ \mathit{true}\,,$$

where $x \in \mathcal{F}$ and $D \in \mathcal{D}$ and a is bound in the scope of $\exists$. The attributed-products of Ψ are the pairs (p, v) where $p \subseteq \mathcal{F}$, $v : \alpha^{-1}(p) \to \cup\mathcal{D}$ satisfies $v \propto \delta$, such that the formula ψ is satisfied by applying in order the next steps: (i) assign *true* to the feature-variables x in p, and assign *false* to the feature-variables in $\mathcal{F} \setminus p$; (ii) assign $v(a)$ to the attribute-variables a in $\alpha^{-1}(p)$, and assign $\mathtt{d}(\delta(a))$ to the others; and (iii) replace each occurrence of $\exists a{:}D.\psi'$ (where ψ' does not contain any further existential, until no occurrence of existentially quantified formula remains) with *true* if there exists n in D such that $\psi'[a := n]$ is satisfied, and with *false* otherwise.

Example 2.10 (A logical representation of the glibc AFM). The extensional representation of the *glibc* AFM given in Example 2.7 has a logical equivalent $\Psi_{\mathsf{glibc}} = (\mathcal{F}_{\mathsf{glibc}}, \mathcal{A}_{\mathsf{glibc}}, \alpha_{\mathsf{glibc}}, \mathcal{D}_{\mathsf{glibc}}, \delta_{\mathsf{glibc}}, \mathtt{d}_{\mathsf{glibc}}, \psi_{\mathsf{glibc}})$ where $\mathcal{F}_{\mathsf{glibc}}$, $\mathcal{A}_{\mathsf{glibc}}$,

α_{glibc}, $\mathcal{D}_{\mathsf{glibc}}$ and δ_{glibc} are defined as in Example 2.7, where ψ_{glibc} is the formula $\phi_{\mathsf{glibc}} \wedge \mathsf{hw}$ where ϕ_{glibc} is given in Example 2.2, and where:

$$\mathsf{d}_{\mathsf{glibc}} = [d_{\mathsf{hw}} \mapsto \text{amd64}, d_{\mathsf{glibc}} \mapsto 2.33].$$

Here, we state as default the most common hardware architecture for personal computers (i.e., a PC 64bits), and the current stable release of *glibc*.

Definition 2.11 (Empty, void and trivial AFMs). The *empty* AFM is $\mathbb{A}_\emptyset = (\emptyset, \emptyset, \perp, \mathcal{D}, \perp, \{(\emptyset, \perp)\})$ — it has no features and has just the empty attributed-product $(\emptyset, \perp)$, where $\perp$ is the function with the empty domain. An AFM is *void* if it has no products, i.e., it is of the form $\mathbf{void}(\mathcal{F}, \mathcal{A}, \alpha, \mathcal{D}, \delta) = (\mathcal{F}, \mathcal{A}, \alpha, \mathcal{D}, \delta, \emptyset)$, for some $\mathcal{F}$, $\mathcal{A}$, α, $\mathcal{D}$ and δ. An AFM is *trivial* if has all the possible products, i.e., it is of the form $\mathbf{trivial}(\mathcal{F}, \mathcal{A}, \alpha, \mathcal{D}, \delta) = (\mathcal{F}, \mathcal{A}, \alpha, \mathcal{D}, \delta, \mathbf{all}(\mathcal{F}, \mathcal{A}, \alpha, \mathcal{D}, \delta))$, where $\mathbf{all}(\mathcal{F}, \mathcal{A}, \alpha, \mathcal{D}, \delta) = \{(p, v) \in 2^{\mathcal{F}} \times \alpha^{-1}(p) \to \cup\mathcal{D} \mid v \propto \delta\}$. Note that $\mathbb{A}_\emptyset$ is trivial.

The logical representations of $\mathbf{void}(\mathcal{F}, \mathcal{A}, \alpha, \mathcal{D}, \delta)$ is $(\mathcal{F}, \mathcal{A}, \alpha, \mathcal{D}, \delta, \mathsf{d}, \mathsf{false})$ for all d. Likewise, $\mathbf{trivial}(\mathcal{F}, \mathcal{A}, \alpha, \mathcal{D}, \delta)$ is logically represented by $(\mathcal{F}, \mathcal{A}, \alpha, \mathcal{D}, \delta, \mathsf{d}, \mathsf{true})$.

2.3. FM Composition and Fragment Relation

Complex software systems, like the Gentoo source-based Linux distribution [20], often consist of many interdependent configurable packages [31, 8, 19]. The configuration options of each package can be represented by a FM. Therefore, configuring two packages in such a way that they can be installed together corresponds to finding a product in the composition of their associated FMs. As pointed out by Lienhardt *et al.* [19], in the logical representation of FMs this composition corresponds to conjunction: the composition of two FMs $(\mathcal{F}_1, \phi_1)$ and $(\mathcal{F}_2, \phi_2)$ is the FM $(\mathcal{F}_1 \cup \mathcal{F}_2, \phi_1 \wedge \phi_2)$. Lienhardt *et al.* [19] also claimed that in the extensional representation of FMs this composition corresponds to the binary operator $\bullet$ of Schröter *et al.* [18], which combines the products sets in way similar to the join operator from relational algebra [32].

Definition 2.12 (FM composition). The composition of two FMs $\mathbb{F}_1 = (\mathcal{F}_1, \mathcal{P}_1)$ and $\mathbb{F}_2 = (\mathcal{F}_2, \mathcal{P}_2)$, denoted $\mathbb{F}_1 \bullet \mathbb{F}_2$, is the FM $(\mathcal{F}_1 \cup \mathcal{F}_2, \{p \cup q \mid p \in \mathcal{P}_1, q \in \mathcal{P}_2, p \cap \mathcal{F}_2 = q \cap \mathcal{F}_1\})$.

As proved in [33, 34], the composition operator $\bullet$ is associative and commutative, with $\mathbb{F}_\emptyset$ as identity element (i.e., $\mathbb{F} \bullet \mathbb{F}_\emptyset = \mathbb{F}$). Note that $(\mathcal{F}_1, \mathcal{P}_1) \bullet (\mathcal{F}_2, \emptyset) = (\mathcal{F}_1 \cup \mathcal{F}_2, \emptyset)$.

Example 2.13 (Composing glibc and gnome-shell FMs). Let us consider another important package of the Gentoo distribution: *gnome-shell*, a core component of the Gnome Desktop environment. Current versions of *gnome-shell* are provided by the package gnome-base/gnome-shell (abbreviated to g-shell in the sequel), and its dependencies include the following statement:

networkmanager?(sys−libs/timezone−data)

This dependency expresses that g-shell requires any version of the time zone database when the feature networkmanager (abbreviated to g-shell:nm in the sequel) is selected.

The *logical representation* of this dependency can be captured by the FM $(\mathcal{F}_{\text{g-shell}}, \phi_{\text{g-shell}})$, where

$$\mathcal{F}_{\text{g-shell}} = \{\text{g-shell, tzdata, g-shell:nm}\} \qquad \phi_{\text{g-shell}} = \text{g-shell} \wedge (\text{g-shell:nm} \rightarrow \text{tzdata})$$

The corresponding *extensional representation* of this FM is $\mathbb{F}_{\text{g-shell}} = (\mathcal{F}_{\text{g-shell}}, \mathcal{P}_{\text{g-shell}})$, where:

$$\mathcal{P}_{\text{g-shell}} = \{\{\text{g-shell}\}, \{\text{g-shell, tzdata}\}, \{\text{g-shell, tzdata, g-shell:nm}\}\}$$

Here, the first two products do not includeg-shell:nm, thus tzdata can be freely selected; in the last product the flag g-shell:nm is selected and tzdata becomes mandatory.

The *logical representation* of the composition is the FM $(\mathcal{F}_{\text{full}}, \phi_{\text{full}})$, where

$$\begin{aligned}
\mathcal{F}_{\text{full}} &= \mathcal{F}_{\text{glibc}} \cup \mathcal{F}_{\text{g-shell}} = \{\text{glibc, texinfo, tzdata, g-shell, glibc:doc, glibc:v, g-shell:nm}\} \\
\phi_{\text{full}} &= \phi_{\text{glibc}} \wedge \phi_{\text{g-shell}} = (\text{glibc} \wedge ((\text{glibc:doc} \rightarrow \text{texinfo}) \wedge (\text{glibc:v} \rightarrow (\neg\text{tz-data}))) \\
&\qquad \wedge(\text{g-shell} \wedge (\text{g-shell:nm} \rightarrow \text{tzdata}))
\end{aligned}$$

The *extensional representation* of the composition is the FM $\mathbb{F}_{\text{full}} = \mathbb{F}_{\text{glibc}} \bullet \mathbb{F}_{\text{g-shell}} = (\mathcal{F}_{\text{full}}, \mathcal{P}_{\text{full}})$ where

$$\begin{aligned}
\mathcal{P}_{\text{full}} = \{&\{\text{glibc, g-shell}\} \cup p \mid p \in 2^{\{\text{texinfo, tzdata}\}}\} \cup \\
&\{\{\text{glibc, glibc:doc, texinfo, g-shell}\} \cup p \mid p \in 2^{\{\text{tzdata}\}}\} \cup \\
&\{\{\text{glibc, glibc:v, g-shell}\} \cup p \mid p \in 2^{\{\text{texinfo}\}}\} \cup \\
&\{\{\text{glibc, g-shell, g-shell:nm, tzdata}\} \cup p \mid p \in 2^{\{\text{texinfo}\}}\} \cup \\
&\{\{\text{glibc, glibc:doc, glibc:v, texinfo, g-shell}\}\} \cup \\
&\{\{\text{glibc, glibc:doc, texinfo, g-shell, g-shell:nm, tzdata}\}\}
\end{aligned}$$

Here, the first line contains the products where both glibc and g-shell are installed, but without use flags selected, so all optional package can be freely selected; the second line contains the products with the glibc's use flag doc selected, so sys-apps/texinfo becomes mandatory; the third line contains the products with the glibc's use flag vanilla selected, so sys-libs/timezone-data is forbidden; the fourth line contains the products with the g-shell's use flag networkmanager, so sys-libs/timezone-data is mandatory; the fifth line contains the product with glibc's both use flags selected and the sixth line contains the product with glibc's use flag doc and g-shell's use flag networkmanager are selected.

The notion of FM composition induces the definition of the notion of FM fragment as a binary relation between FMs.

Definition 2.14 (FM fragment relation). A FM $\mathbb{F}_0$ is a *fragment* of a FM $\mathbb{F}_1$, denoted as $\mathbb{F}_0 \leq \mathbb{F}_1$, whenever there exists a FM $\mathbb{F}'$ such that $\mathbb{F}_0 \bullet \mathbb{F}' = \mathbb{F}_1$.

For instance, we have (by definition) that $\mathbb{F}_{\text{g-shell}} \leq (\mathbb{F}_{\text{g-shell}} \bullet \mathbb{F}_{\text{glibc}})$. It is worth observing that, as illustrated by the following example, some combination of features that are allowed in the members of the composition might be no longer available in the result of the composition.

Example 2.15 (Composing glibc and libical FMs). Consider for instance the library libical in Gentoo. Its FM contains the following constraint (as expressed in Gentoo notation):

berkdb? (sys−libs/db) sys−libs/timezone−data

This dependency expresses that libical requires the db library whenever the feature berkdb is selected and requires the package sys-libs/timezone-data to be installed. These *dependencies* can be extensionally expressed by a FM $\mathbb{F}_{\text{libical}} = (\mathcal{F}_{\text{libical}}, \mathcal{P}_{\text{libical}})$ where

$$\mathcal{F}_{\text{libical}} = \{\text{libical, berkdb, sys-libs/db, tzdata}\}$$
$$\mathcal{P}_{\text{libical}} = \{\{\text{libical, tzdata}\}, \{\text{libical, tzdata, berkdb, sys-libs/db}\}\}$$

Composing the FM of glibc and libical gives the FM $\mathbb{F}_c = (\mathcal{F}_c, \mathcal{P}_c)$ where $\mathcal{F}_c = \mathcal{F}_{\text{glibc}} \cup \mathcal{F}_{\text{libical}}$ and:

$$\begin{aligned}\mathcal{P}_c = \{&\{\text{glibc, libical, tzdata}\} \cup p \mid p \in 2^{\{\text{texinfo, sys-libs/db}\}}\} \cup \\ &\{\{\text{glibc, glibc:doc, texinfo, libical, tzdata}\} \cup p \mid p \in 2^{\{\text{sys-libs/db}\}}\} \cup \\ &\{\{\text{glibc, libical, berkdb, sys-libs/db, tzdata}\} \cup p \mid p \in 2^{\{\text{texinfo}\}}\} \cup \\ &\{\{\text{glibc, glibc:doc, texinfo, libical, berkdb, sys-libs/db, tzdata}\}\}\end{aligned}$$

Here, the first line contains the products where both glibc and libical are installed, but without use flags selected, so only the annex package timezone-data is mandatory; the second line contains the products with the glibc's use flag doc selected, so sys-apps/texinfo becomes mandatory; the contains the products with the libical's use flag berkdb, so sys-libs/db becomes mandatory; finally, the fourth line contains the product with all optional features of both glibc and libical selected.

It is easy to see from the constraint, and also from the extensional representation, that combining glibc and libical makes the feature glibc:v dead (i.e., not selectable): when composed, the FMs interact and not all combinations of products are available. Feature incompatibilities such as this are a normal occurrence in many product lines (such as the linux kernel) but have two negative properties: first, it means that some features that are stated to be optional (i.e., can be freely selected or not by the user) actually are not optional in some cases, depending on some other packages being selected or not; second, it means that some packages cannot be installed at the same time because of their dependencies: consider for instance a package that requires the feature glibc:v being selected, that package is not compatible with libical.

2.4. AFM Composition and Fragment Relation

Notation 2.16. If v_1, v_2 are functions then we write $v_1 \Leftrightarrow v_2$ to mean that they coincide on their common domain, i.e., $v_1(x) = v_2(x)$, for all $x \in \text{dom}(v_1) \cap$

$\mathrm{dom}(v_2)$. If $v_1 \eqcirc v_2$ then v_1, v_2 are compatible, so we can define $v_1 \oplus v_2$ be the function with domain $\mathrm{dom}(v_1) \cup \mathrm{dom}(v_2)$ behaving as v_1 on $\mathrm{dom}(v_1)$ and v_2 on $\mathrm{dom}(v_2)$.

Definition 2.17 (AFM compatibility and composition). Let $\mathbb{A}_i = (\mathcal{F}_i, \mathcal{A}_i, \alpha_i, \mathcal{D}_i, \delta_i, \mathcal{V}_i)$ be AFM where $i = 0, 1$. If $\alpha_0 \eqcirc \alpha_1$, $\delta_0 \eqcirc \delta_1$ then $\mathbb{A}_0, \mathbb{A}_1$ are said *compatible*. If $\mathbb{A}_0, \mathbb{A}_1$ are compatible then, we write $\mathbb{A}_0 \bullet \mathbb{A}_1$ for their composition, namely the AFM $(\mathcal{F}_0 \cup \mathcal{F}_1, \mathcal{A}_0 \cup \mathcal{A}_1, \alpha_0 \oplus \alpha_1, \mathcal{D}_0 \cup \mathcal{D}_1, \delta_0 \oplus \delta_1, \mathcal{V})$ such that

$$\mathcal{V} = \{(p_0 \cup p_1, v_0 \oplus v_1) \mid (p_i, v_i) \in \mathcal{V}_i \ , \ p_0 \cap \mathcal{F}_1 = p_1 \cap \mathcal{F}_0 \ , \ v_0 \eqcirc v_1\} .$$

In the following, we reason under the hypothesis that each attribute uniquely identifies the feature to which it is associated (cf. Example 2.8), thus we focus on compatible AFMs. Whenever the compatibility holds, it follows that $\bullet$ is associative and commutative also for AFM; following the proofs given in [33, 34]. As for traditional FMs, each AFM $\mathbb{A}$ is idempotent w.r.t. the composition, viz. $\mathbb{A} \bullet \mathbb{A} = \mathbb{A}$. Moreover, note that $((\mathcal{F}_1, \mathcal{P}_1), \mathcal{A}_1, \mathcal{D}_1, \alpha_1, \mathcal{V}_1) \bullet ((\mathcal{F}_2, \emptyset), \mathcal{A}_2, \mathcal{D}_2, \alpha_2, \emptyset) = ((\mathcal{F}_1 \cup \mathcal{F}_2, \emptyset)\mathcal{A}_1 \cup \mathcal{A}_2, \mathcal{D}_1 \cup \mathcal{D}_2, \alpha_1 \oplus \alpha_2, \emptyset)$ and that $\mathbb{A}_\emptyset$ is the identity for $\bullet$.

Example 2.18 (Composing glibc and gnome-shell AFMs). Like for *glibc*, the *gnome-shell* package has several versions and requirement on the hardware architecture it can be installed on. To model these aspects, we extend the $\mathbb{F}_{\mathsf{g\text{-}shell}}$ FM into a AFM with: the feature for the hardware configuration hw the $\mathsf{keyword}$ attribute and an attribute $\mathsf{g\text{-}shell}_v$ for the version of the *gnome-shell* package. This extension results into the AFM $\mathbb{A}_{\mathsf{g\text{-}shell}} = (\mathcal{F}'_{\mathsf{g\text{-}shell}}, \mathcal{A}_{\mathsf{g\text{-}shell}}, \alpha_{\mathsf{g\text{-}shell}}, \mathcal{D}_{\mathsf{g\text{-}shell}}, \delta_{\mathsf{g\text{-}shell}}, \mathcal{V}_{\mathsf{g\text{-}shell}})$ where:

$$\begin{aligned}
&\mathcal{F}'_{\mathsf{g\text{-}shell}} = \mathcal{F}_{\mathsf{g\text{-}shell}} \cup \{\mathsf{hw}\}, \quad \mathcal{A}_{\mathsf{g\text{-}shell}} = \{\mathsf{g\text{-}shell}_v, \mathsf{keyword}\}, \quad \mathcal{D}_{\mathsf{g\text{-}shell}} = \{d_{\mathsf{g\text{-}shell}}, d_{\mathsf{hw}}\},\\
&\alpha_{\mathsf{g\text{-}shell}} = [\mathsf{g\text{-}shell}_v \mapsto \mathsf{g\text{-}shell}, \mathsf{keyword} \mapsto \mathsf{hw}],\\
&\delta_{\mathsf{g\text{-}shell}} = [\mathsf{g\text{-}shell}_v \mapsto d_{\mathsf{g\text{-}shell}}, \mathsf{keyword} \mapsto d_{\mathsf{hw}}],\\
&\mathcal{V}_{\mathsf{g\text{-}shell}} = \left\{ (p \cup \{\mathsf{hw}\}, [\mathsf{keyword} \mapsto k, \mathsf{g\text{-}shell}_v \mapsto 3.36.7]) \;\middle|\; \begin{array}{l} p \in \mathcal{P}_{\mathsf{g\text{-}shell}},\ \mathsf{g\text{-}shell} \in p,\\ k \in d_{\mathsf{hw}} \end{array} \right\}\\
&\qquad \cup \left\{ (p \cup \{\mathsf{hw}\}, [\mathsf{keyword} \mapsto k, \mathsf{g\text{-}shell}_v \mapsto v]) \;\middle|\; \begin{array}{l} p \in \mathcal{P}_{\mathsf{g\text{-}shell}},\ \mathsf{g\text{-}shell} \in p,\\ k \in d^2_{\mathsf{hw}}, v \in d^1_{\mathsf{g\text{-}shell}} \end{array} \right\}
\end{aligned}$$

with $d_{\mathsf{g\text{-}shell}} = \{3.36.7, 3.38.4, 3.38.4\text{-r1}\}$, $d_{\mathsf{hw}} = \{\text{amd64, x86, arm, arm64, ia64, ppc, ppc64}\}$, $d^2_{\mathsf{hw}} = \{\text{amd64, x86, arm, arm64, ppc64}\}$, $d^1_{\mathsf{g\text{-}shell}} = \{3.38.4, 3.38.4\text{-r1}\}$.

Here, we can see that not all versions of *gnome-shell* can be installed on every architecture: all versions can be installed on amd64, x86, arm, arm64 and ppc64, and only version 3.36.7 can be additionally installed on the architectures ia64 and ppc.

It is clear that $\mathbb{A}_{\mathsf{glibc}}$ and $\mathbb{A}_{\mathsf{g\text{-}shell}}$ are *compatible*: the only shared attribute is $\mathsf{keyword}$ which is linked in $\mathbb{A}_{\mathsf{glibc}}$ and $\mathbb{A}_{\mathsf{g\text{-}shell}}$ to the same feature hw and the same domain d_{hw}. By Definition 2.17, we can thus construct the AFM $\mathbb{A}_{\mathsf{full}} = \mathbb{A}_{\mathsf{glibc}} \bullet \mathbb{A}_{\mathsf{g\text{-}shell}}$. By construction, we thus have $\mathbb{A}_{\mathsf{full}} = (\mathcal{F}'_{\mathsf{full}}, \mathcal{A}_{\mathsf{full}}, \alpha_{\mathsf{full}}, \mathcal{D}_{\mathsf{full}}, \delta_{\mathsf{full}}, \mathcal{V}_{\mathsf{full}})$

where:

$$
\begin{aligned}
&\mathcal{F}'_{\mathsf{full}} = \mathcal{F}'_{\mathsf{glibc}} \cup \mathcal{F}'_{\mathsf{g\text{-}shell}} = \mathcal{F}_{\mathsf{full}} \cup \{\mathsf{hw}\} \\
&\mathcal{A}_{\mathsf{full}} = \mathcal{A}_{\mathsf{glibc}} \cup \mathcal{A}_{\mathsf{g\text{-}shell}} = \{\mathsf{g\text{-}shell}_v, \mathsf{glibc}_v, \mathsf{keyword}\} \\
&\alpha_{\mathsf{full}} = [\mathsf{glibc}_v \mapsto \mathsf{glibc}, \mathsf{g\text{-}shell}_v \mapsto \mathsf{g\text{-}shell}, \mathsf{keyword} \mapsto \mathsf{hw}] \\
&\mathcal{D}_{\mathsf{full}} = \mathcal{D}_{\mathsf{glibc}} \cup \mathcal{D}_{\mathsf{g\text{-}shell}} = \{d_{\mathsf{glibc}}, d_{\mathsf{g\text{-}shell}}, d_{\mathsf{hw}}\} \\
&\delta_{\mathsf{full}} = [\mathsf{glibc}_v \mapsto d_{\mathsf{glibc}}, \mathsf{g\text{-}shell}_v \mapsto d_{\mathsf{g\text{-}shell}}, \mathsf{keyword} \mapsto d_{\mathsf{hw}}] \\
&\mathcal{V}_{\mathsf{full}} = \left\{ \left(p \cup p', \begin{bmatrix} \mathsf{keyword} \mapsto k, \\ \mathsf{glibc}_v \mapsto v, \\ \mathsf{g\text{-}shell}_v \mapsto 3.36.7 \end{bmatrix} \right) \middle| \begin{array}{l} p \in \mathcal{P}_{\mathsf{glibc}},\ \mathsf{glibc} \in p, \\ p' \in \mathcal{P}_{\mathsf{g\text{-}shell}},\ \mathsf{g\text{-}shell} \in p', \\ v \in d_{\mathsf{glibc}},\ k \in d_{\mathsf{hw}} \end{array} \right\} \\
&\qquad \bigcup \left\{ \left(p \cup p', \begin{bmatrix} \mathsf{keyword} \mapsto k, \\ \mathsf{glibc}_v \mapsto v, \\ \mathsf{g\text{-}shell}_v \mapsto v' \end{bmatrix} \right) \middle| \begin{array}{l} p \in \mathcal{P}_{\mathsf{glibc}},\ \mathsf{glibc} \in p, \\ p' \in \mathcal{P}_{\mathsf{g\text{-}shell}},\ \mathsf{g\text{-}shell} \in p', \\ v \in d_{\mathsf{glibc}},\ v' \in d^1_{\mathsf{g\text{-}shell}},\ k \in d^2_{\mathsf{hw}} \end{array} \right\}
\end{aligned}
$$

Most of the elements defining $\mathbb{A}_{\mathsf{full}}$ are simply the union of the elements of $\mathbb{A}_{\mathsf{glibc}}$ and $\mathbb{A}_{\mathsf{g\text{-}shell}}$. The interesting exception is $\mathcal{V}_{\mathsf{full}}$: the first line includes all the attributed-products with glibc and version 3.36.7 of the *gnome-shell* package; and the second line includes all the attributed-products with glibc and the other versions of the *gnome-shell* package. We can see that all of the hardware restrictions required by the different versions of the *gnome-shell* package are also enforced in the attributed-products of $\mathbb{A}_{\mathsf{full}}$.

The notion of fragment relation can be straight adapted to AFM.

Definition 2.19 (AFM fragment relation). Let $\mathbb{A}_0, \mathbb{A}_1$ be compatible AFM. A FM $\mathbb{A}_0$ is a *fragment* of a FM $\mathbb{A}_1$, denoted as $\mathbb{A}_0 \leq \mathbb{A}_1$, whenever there exists a FM $\mathbb{A}'$ such that $\mathbb{A}_0 \bullet \mathbb{A}' = \mathbb{A}_1$.

Let $\mathbb{A}_i = (\mathcal{F}_i, \mathcal{A}_i, \alpha_i, \mathcal{D}_i, \delta_i, \mathcal{V}_i)$ be two AFMs where $i = 0, 1$: if $\mathbb{A}_0 \leq \mathbb{A}_1$ then $(\mathcal{F}_0, \mathcal{P}_{\mathcal{V}_0}) \leq (\mathcal{F}_1, \mathcal{P}_{\mathcal{V}_1})$ holds.

Example 2.20 (Composing glibc, gnome-shell and libical AFMs). We already saw in Example 2.18 that even if keyword is allowed to take all of its possible values in $\mathbb{A}_{\mathsf{glibc}}$ when selecting glibc, when composing $\mathbb{A}_{\mathsf{glibc}}$ with $\mathbb{A}_{\mathsf{g\text{-}shell}}$ and selecting g-shell, keyword can now only take the values in d^2_{hw} (or d_{hw} when version 3.36.7 of *gnome-shell* is selected). Hence the removal of feature combinations due to composition also occur for the domains of attributes.

We can combine both effects by composing the AFM of *glibc*, *gnome-shell* and *libical* together. Currently, the *libical* package has three available versions in Gentoo, that are available on every architecture: we define the AFM of *libical* as $\mathbb{A}_{\mathsf{libical}} = (\mathcal{F}'_{\mathsf{libical}}, \mathcal{A}_{\mathsf{libical}}, \alpha_{\mathsf{libical}}, \mathcal{D}_{\mathsf{libical}}, \delta_{\mathsf{libical}}, \mathcal{V}_{\mathsf{libical}})$ where:

$$
\begin{aligned}
&\mathcal{F}'_{\mathsf{libical}} = \mathcal{F}_{\mathsf{libical}} \cup \{\mathsf{hw}\} \\
&\mathcal{A}_{\mathsf{libical}} = \{\mathsf{libical}_v, \mathsf{keyword}\} \\
&\alpha_{\mathsf{libical}} = [\mathsf{libical}_v \mapsto \mathsf{libical}, \mathsf{keyword} \mapsto \mathsf{hw}] \\
&\mathcal{D}_{\mathsf{libical}} = \{d_{\mathsf{libical}}, d_{\mathsf{hw}}\} \\
&\delta_{\mathsf{libical}} = [\mathsf{libical}_v \mapsto d_{\mathsf{libical}}, \mathsf{keyword} \mapsto d_{\mathsf{hw}}] \\
&\mathcal{V}_{\mathsf{libical}} = \{(p \cup \{\mathsf{hw}\}, [\mathsf{keyword} \mapsto k, \mathsf{libical}_v \mapsto v]) \mid p \in \mathcal{P}_{\mathsf{libical}},\ \mathsf{libical} \in p,\ k \in d_{\mathsf{hw}}\}
\end{aligned}
$$

with $d_{\mathsf{libical}} = \{3.0.8, 3.0.9, 3.0.10\}$.
Composing $\mathbb{A}_{\mathsf{full}}$ with $\mathbb{A}_{\mathsf{libical}}$ will thus result in an AFM in which, when g-shell and libical are selected, keyword being restricted to the values in d^2_{hw} (or d_{hw} depending on the version of g-shell), and where the feature tzdata is mandatory. We thus define $\mathbb{A}_{\mathsf{all}} = \mathbb{A}_{\mathsf{full}} \bullet \mathbb{A}_{\mathsf{libical}} = (\mathcal{F}_{\mathsf{all}}, \mathcal{A}_{\mathsf{all}}, \alpha_{\mathsf{all}}, \mathcal{D}_{\mathsf{all}}, \delta_{\mathsf{all}}, \mathcal{V}_{\mathsf{all}})$ where:

$$
\begin{aligned}
&\mathcal{F}_{\mathsf{all}} = \mathcal{F}'_{\mathsf{full}} \cup \mathcal{F}_{\mathsf{libical}}\\
&\mathcal{A}_{\mathsf{all}} = \mathcal{A}_{\mathsf{full}} \cup \mathcal{A}_{\mathsf{libical}} = \{\mathsf{glibc}_v, \mathsf{g\text{-}shell}_v, \mathsf{libical}_v, \mathsf{keyword}\}\\
&\alpha_{\mathsf{all}} = [\mathsf{glibc}_v \mapsto \mathsf{glibc}, \mathsf{g\text{-}shell}_v \mapsto \mathsf{g\text{-}shell}, \mathsf{libical}_v \mapsto \mathsf{libical}, \mathsf{keyword} \mapsto \mathsf{hw}]\\
&\mathcal{D}_{\mathsf{all}} = \{d_{\mathsf{glibc}}, d_{\mathsf{g\text{-}shell}}, d_{\mathsf{libical}}, d_{\mathsf{hw}}\}\\
&\delta_{\mathsf{all}} = [\mathsf{glibc}_v \mapsto d_{\mathsf{glibc}}, \mathsf{g\text{-}shell}_v \mapsto d_{\mathsf{g\text{-}shell}}, \mathsf{libical}_v \mapsto d_{\mathsf{libical}}, \mathsf{keyword} \mapsto d_{\mathsf{hw}}]\\
&\mathcal{V}_{\mathsf{all}} = \{(p \cup p', v \cup [\mathsf{libical}_v \mapsto v']) \mid (p, v) \in \mathcal{V}_{\mathsf{full}},\ p' \in \mathcal{P}_{\mathsf{libical}}, \mathsf{libical} \in p',\ v' \in d_{\mathsf{libical}}\}.
\end{aligned}
$$

3. Algebraic Characterization of FMs and AFMs

In Section 3.1, we recall some relevant algebraic notions. In Section 3.2, we show that the FM fragment relation induces a lattice of FMs where the join operation is FM composition, and in Section 3.3 we extend the result to AFMs. Then, in Section 3.4, we show that the FM fragment relation generalizes the FM interface relation [18] and we provide some more algebraic properties, and in Section 3.5 we extend these results to AFMs.

3.1. A Recollection of Algebraic Notions

In this section we briefly recall the notions of lattice, bounded lattice and Boolean algebra (see, e.g., Davey and Priestley [35] for a detailed presentation). An *ordered lattice* is a partially ordered set $(P, \sqsubseteq)$ such that, for every $x, y \in P$, both the least upper bound (lub) of $\{x, y\}$, denoted $\sup\{x, y\} = \min\{a \mid x, y \sqsubseteq a\}$, and the greatest lower bound (glb) of $\{x, y\}$, denoted $\inf\{x, y\} = \max\{a \mid a \sqsubseteq x, y\}$, are always defined.

An *algebraic lattice* is an algebraic structure $(L, \sqcup, \sqcap)$ where L is non-empty set equipped with two binary operations $\sqcup$ (called *join*) and $\sqcap$ (called *meet*) which satisfy the following.

- Associative laws: $x \sqcup (y \sqcup z) = (x \sqcup y) \sqcup z$, $x \sqcap (y \sqcap z) = (x \sqcap y) \sqcap z$.
- Commutative laws: $x \sqcup y = y \sqcup x$, $x \sqcap y = y \sqcap x$.
- Absorption laws: $x \sqcup (x \sqcap y) = x$, $x \sqcap (x \sqcup y) = x$.
- Idempotency laws: $x \sqcup x = x$, $x \sqcup x = x$.

As known, the two notions of lattice are equivalent (Theorem 2.9 and 2.10 of [35]). In particular, given an ordered lattice $(P, \sqsubseteq)$ with the operations $x \sqcup y = \sup\{x, y\}$ and $x \sqcap y = \inf\{x, y\}$, the following three statements are equivalent (Theorem 2.8 of [35]):

$$x \sqsubseteq y, \qquad x \sqcup y = y, \qquad x \sqcap y = x.$$

A *bounded lattice* is a lattice that contains two elements $\bot$ (the lattice's *bottom*) and $\top$ (the lattice's *top*) which satisfy the following law: $\bot \sqsubseteq x \sqsubseteq \top$. Let L be a bounded lattice, $y \in L$ is a *complement* of $x \in L$ if $x \sqcap y = \bot$ and $x \sqcup y = \top$. If x has a unique complement, we denote this complement by $x^{\complement}$.

A *distributive lattice* is a lattice which satisfies the following distributive law: $x \sqcap (y \sqcup z) = (x \sqcap y) \sqcup (x \sqcap z)$. In a bounded distributive lattice the complement (whenever it exists) is unique (see [35, Section 4.13]).

A *Boolean lattice* (a.k.a. *Boolean algebra*) L is a bounded distributive lattice such that each $x \in L$ has a (necessarily unique) complement $x^{\complement} \in L$.

3.2. Lattices of FMs

Although (to the best of our knowledge) only finite FMs are relevant in practice, in our theoretical development (in order to enable a better understanding of the relation between the extensional and the logical representations) we consider also FMs with infinitely many features and products. The following definition introduces a notation for three different sets of extensional FMs (see Definition 2.3) over a given set of features.

Definition 3.1 (Sets of extensional FMs over a set of features). Let X be a set of features. We denote:

- $\mathfrak{E}(X)$ the set of the extensional FMs $(\mathcal{F}, \mathcal{P})$ such that $\mathcal{F} \subseteq X$;
- $\mathfrak{E}_{\text{fin}}(X)$ the set of $(\mathcal{F}, \mathcal{P}) \in \mathfrak{E}(X)$ such that $\mathcal{F} \subseteq_{\text{fin}} X$; and
- $\mathfrak{E}_{\text{eql}}(X)$ the set of $(\mathcal{F}, \mathcal{P}) \in \mathfrak{E}(X)$ such that $\mathcal{F} = X$.

Note that, if X has infinitely many elements then $\mathfrak{E}_{\text{fin}}(X)$ has infinitely many elements too. Instead, if X is finite then $\mathfrak{E}(X)$ and $\mathfrak{E}_{\text{fin}}(X)$ coincide and have a finite number of elements.

The FMs used in practice are finite, i.e., they belong to $\mathfrak{E}_{\text{fin}}(X)$. However, as we will see, $\mathfrak{E}_{\text{fin}}(X)$ is algebraically more poor that the other two sets of FMs. In particular, $\mathfrak{E}_{\text{eql}}(X)$ is the algebraically richer structure.

Notation 3.2. Let $\mathcal{F}$ be a set of feature. If $X \subseteq \mathcal{F}$ and $Y \subseteq 2^{\mathcal{F}}$ then, we write $Y|_X$ to denote $\{y \cap X \mid y \in Y\}$.

Lemma 3.3 (Two criteria for the FM fragment relation). *For all* $\mathbb{F}_1 = (\mathcal{F}_1, \mathcal{P}_1) \in \mathfrak{E}(X)$ *and* $\mathbb{F}_2 = (\mathcal{F}_2, \mathcal{P}_2) \in \mathfrak{E}(X)$*, the following statements are equivalent:*

i) $\mathbb{F}_1 \leq \mathbb{F}_2$*;*

ii) $\mathbb{F}_1 \bullet \mathbb{F}_2 = \mathbb{F}_2$*;*

iii) $\mathcal{F}_1 \subseteq \mathcal{F}_2$ *and* $\mathcal{P}_1 \supseteq \mathcal{P}_2|_{\mathcal{F}_1}$*.*

Proof. *i)* $\Rightarrow$ *ii)*. It is straightforward to check that $\mathbb{F} \bullet \mathbb{F} = \mathbb{F}$, for all $\mathbb{F}$. Then, by definition of $\leq$ (Definition 2.14) there is $\mathbb{F}' \in \mathfrak{C}(X)$ such that $\mathbb{F}_2 = \mathbb{F}_1 \bullet \mathbb{F}'$. Thus, $\mathbb{F}_1 \bullet \mathbb{F}_2 = \mathbb{F}_1 \bullet (\mathbb{F}_1 \bullet \mathbb{F}') = (\mathbb{F}_1 \bullet \mathbb{F}_1) \bullet \mathbb{F}' = \mathbb{F}_1 \bullet \mathbb{F}' = \mathbb{F}_2$.
ii) $\Rightarrow$ *iii)*. By definition of $\bullet$ (Definition 2.12), it is clear from the hypothesis that $\mathcal{F}_1 \subseteq \mathcal{F}_2$. Moreover, $\mathcal{P}_2 = \{p \cup q \mid p \in \mathcal{P}_1, q \in \mathcal{P}_2, p \cap \mathcal{F}_2 = q \cap \mathcal{F}_1\}$ immediately implies that $\mathcal{P}_2 = \{q \mid p \in \mathcal{P}_1, q \in \mathcal{P}_2, p = q \cap \mathcal{F}_1\}$, which in turn implies $\mathcal{P}_2|_{\mathcal{F}_1} \subseteq \mathcal{P}_1$.
iii) $\Rightarrow$ *i)*. By using the hypothesis, we have $(\mathcal{F}_1 \cup \mathcal{F}_2, \{p \cup q \mid p \in \mathcal{P}_1,\, q \in \mathcal{P}_2,\, p \cap \mathcal{F}_2 = q \cap \mathcal{F}_1\}) = (\mathcal{F}_2, \mathcal{P}_2)$, i.e. $\mathbb{F}_1 \bullet \mathbb{F}_2 = \mathbb{F}_2$. This implies, by definition of $\leq$, that $\mathbb{F}_1 \leq \mathbb{F}_2$. □

It is worth observing that, if $\mathbb{F}_1 \leq \mathbb{F}_2$, then $\mathcal{P}_1$ can contain products that are not subsets of products in $\mathcal{P}_2$. Clearly, Lemma 3.3 applies to $\mathfrak{C}_{\text{fin}}(X), \mathfrak{C}_{\text{eql}}(X)$ too.

Lemma 3.4 (The operator $\bullet$ on $\mathfrak{C}_{\text{eql}}(X)$). *Let $\mathbb{F}_1 = (\mathcal{F}_1, \mathcal{P}_1), \mathbb{F}_2 = (\mathcal{F}_2, \mathcal{P}_2)$ be FMs in $\mathfrak{C}_{\text{eql}}(X)$, we have that: $\mathbb{F}_1 \bullet \mathbb{F}_2 = (X, \mathcal{P}_1 \cap \mathcal{P}_2)$.*

Proof. According to the definition of $\bullet$ we have:
$\mathbb{F}_1 \bullet \mathbb{F}_2 = (X, \{p_1 \cup p_2 \mid p_1 \in \mathcal{P}_1, p_2 \in \mathcal{P}_2, p_1 = p_2\}) = (X, \mathcal{P}_1 \cap \mathcal{P}_2)$. □

Definition 3.5 (Meet for FMs). Let $\mathbb{F}_1 = (\mathcal{F}_1, \mathcal{P}_1), \mathbb{F}_2 = (\mathcal{F}_2, \mathcal{P}_2)$ be in $\mathfrak{C}(X)$, we define: $\mathbb{F}_1 \star \mathbb{F}_2 = (\mathcal{F}_1 \cap \mathcal{F}_2,\ \mathcal{P}_1|_{\mathcal{F}_2} \cup \mathcal{P}_2|_{\mathcal{F}_1})$.

Lemma 3.6 (The operator $\star$ on $\mathfrak{C}_{\text{eql}}(X)$). *Let $\mathbb{F}_1 = (\mathcal{F}_1, \mathcal{P}_1), \mathbb{F}_2 = (\mathcal{F}_2, \mathcal{P}_2)$ in $\mathfrak{C}_{\text{eql}}(X)$, we have that: $\mathbb{F}_1 \star \mathbb{F}_2 = (X, \mathcal{P}_1 \cup \mathcal{P}_2)$.*

Proof. According to the definition of $\star$ we have:
$\mathbb{F}_1 \star \mathbb{F}_2 = (X \cap X,\ \mathcal{P}_1|_X \cup \mathcal{P}_2|_X) = (X, \mathcal{P}_1 \cup \mathcal{P}_2)$. □

The following definition introduces an operator that plays the role of complement in the structure $\mathfrak{C}_{\text{eql}}(X)$. Namely, $\mathbb{F}^\dagger$ is the FM which has as products the configurations of the FM $\mathbb{F}$ that are not products of $\mathbb{F}$.

Definition 3.7 (Complement on $\mathfrak{C}_{\text{eql}}(X)$). Let $\mathbb{F} = (\mathcal{F}, \mathcal{P})$ be in $\mathfrak{C}(X)$, we define: $\mathbb{F}^\dagger = (\mathcal{F}, 2^{\mathcal{F}} \setminus \mathcal{P})$.

Theorem 3.8 (Lattices of FMs over a set of features).

1. *$(\mathfrak{C}(X), \leq)$ is a bounded lattice with join $\bullet$, meet $\star$, bottom* **trivial**$(\emptyset) = (\emptyset, \{\emptyset\})$ *and top* **void**$(X) = (X, \emptyset)$.

2. *If X is an infinite set then $\mathfrak{C}_{\text{fin}}(X)$ is a sublattice of $\mathfrak{C}(X)$ with the same bottom and no top.*

3. *$\mathfrak{C}_{\text{eql}}(X)$ is a sublattice of $\mathfrak{C}(X)$ and it is a Boolean lattice with bottom* **trivial**$(X) = (X, 2^X)$, *same top of $\mathfrak{C}(X)$, and complement $(\cdot)^\dagger$.*

Proof. Let $\mathbb{F}_1 = (\mathcal{F}_1, \mathcal{P}_1)$ and $\mathbb{F}_2 = (\mathcal{F}_2, \mathcal{P}_2)$ be FMs. First prove that $\leq$ is a partial order. Let $\mathbb{F}_1 \leq \mathbb{F}_2 \leq \mathbb{F}_3$.

- *Reflexivity.* The proof for $\mathfrak{C}(X)$ and $\mathfrak{C}_{\text{fin}}(X)$ follows because $\mathbb{F}_1 \bullet \mathbb{F}_\emptyset = \mathbb{F}_1$. In the case $\mathfrak{C}_{\text{eql}}(X)$, it follows because $\mathbb{F}_1 \bullet (X, 2^X) = \mathbb{F}_1$. Clearly, $\mathbb{F}_\emptyset$ belongs to $\mathfrak{C}_{\text{eql}}(X)$ only when $X = \emptyset$ (in this case, $(X, 2^X)$ is $\mathbb{F}_\emptyset$).

- *Antisymmetry.* The proof is the same for $\mathfrak{C}(X)$, $\mathfrak{C}_{\text{fin}}(X)$, $\mathfrak{C}_{\text{eql}}(X)$. Suppose $\mathbb{F}_2 \leq \mathbb{F}_1$ and $\mathbb{F}_1 \leq \mathbb{F}_2$, so by hypothesis there are $\mathbb{F}, \mathbb{F}'$ such that $\mathbb{F}_1 = \mathbb{F}_2 \bullet \mathbb{F}$ and $\mathbb{F}_2 = \mathbb{F}_1 \bullet \mathbb{F}'$. Clearly, by associativity, commutativity and idempotency

$$\begin{aligned} \mathbb{F}_1 &= \mathbb{F}_2 \bullet \mathbb{F} = (\mathbb{F}_1 \bullet \mathbb{F}') \bullet \mathbb{F} = ((\mathbb{F}_2 \bullet \mathbb{F}) \bullet \mathbb{F}') \bullet \mathbb{F} \\ &= \mathbb{F}_2 \bullet (\mathbb{F} \bullet \mathbb{F}) \bullet \mathbb{F}' = (\mathbb{F}_2 \bullet \mathbb{F}) \bullet \mathbb{F}' = \mathbb{F}_1 \bullet \mathbb{F}' = \mathbb{F}_2 . \end{aligned}$$

- *Transitivity.* The proof is the same for $\mathfrak{C}(X)$, $\mathfrak{C}_{\text{fin}}(X)$, $\mathfrak{C}_{\text{eql}}(X)$. Let $\mathbb{F}, \mathbb{F}'$ such that $\mathbb{F}_3 = \mathbb{F}_2 \bullet \mathbb{F}$ and $\mathbb{F}_2 = \mathbb{F}_1 \bullet \mathbb{F}'$. Clearly, $\mathbb{F}_3 = \mathbb{F}_2 \bullet \mathbb{F} = (\mathbb{F}_1 \bullet \mathbb{F}') \bullet \mathbb{F} = \mathbb{F}_1 \bullet (\mathbb{F}' \bullet \mathbb{F})$ which ensures that $\mathbb{F}_1 \leq \mathbb{F}_3$.

Part 1: $(\mathfrak{C}(X), \leq)$ is a bounded lattice with join $\bullet$, meet $\star$, bottom $\mathbb{F}_\emptyset = (\emptyset, \{\emptyset\})$ and top $(X, \emptyset)$. Let $\uparrow \mathbb{F}$ be the set of upper bounds of $\mathbb{F}$ w.r.t. $\leq$, i.e. $\{\mathbb{F}' \mid \mathbb{F} \leq \mathbb{F}'\}$; and, let $\downarrow \mathbb{F}$ be the set of lower bounds of $\mathbb{F}$ w.r.t. $\leq$, i.e. $\{\mathbb{F}' \mid \mathbb{F}' \leq \mathbb{F}\}$.

- $\mathbb{F}_i \leq \mathbb{F}_1 \bullet \mathbb{F}_2$ $(i = 1, 2)$, by definition of $\leq$; so $\mathbb{F}_1 \bullet \mathbb{F}_2 \in (\uparrow \mathbb{F}_1) \cap (\uparrow \mathbb{F}_2)$. Moreover, for all common upper bounds $\mathbb{F} \in (\uparrow \mathbb{F}_1) \cap (\uparrow \mathbb{F}_2)$, we have, by Lemma 3.3, $\mathbb{F} = \mathbb{F}_1 \bullet \mathbb{F} = \mathbb{F}_1 \bullet (\mathbb{F}_2 \bullet \mathbb{F}) = (\mathbb{F}_1 \bullet \mathbb{F}_2) \bullet \mathbb{F}$. Thus, we conclude that $\mathbb{F}_1 \bullet \mathbb{F}_2$ is the join.

- Let $\mathbb{F} = (\mathcal{F}, \mathcal{P}) = (\mathcal{F}_1 \cap \mathcal{F}_2, \mathcal{P}_1|_{\mathcal{F}_2} \cup \mathcal{P}_2|_{\mathcal{F}_1})$. We have $\{p \cap \mathcal{F} \mid p \in \mathcal{P}_i\} \subseteq \mathcal{P}$ and $\mathcal{F} \subseteq \mathcal{F}_i$ for $i \in \{1, 2\}$: we thus have $\mathbb{F} \in (\downarrow \mathbb{F}_1) \cap (\downarrow \mathbb{F}_2)$. Moreover, for all $(\mathcal{F}', \mathcal{P}') \in (\downarrow \mathbb{F}_1) \cap (\downarrow \mathbb{F}_2)$, it is easy to see that $\mathcal{F}' \subseteq \mathcal{F}_1 \cap \mathcal{F}_2 \subseteq \mathcal{F}$ and $\mathcal{P}|_{\mathcal{F}'} \subseteq \mathcal{P}'$ by Lemma 3.3. And so, again by Lemma 3.3, $\mathbb{F}$ is the meet.

- For all $\mathbb{F} \in \mathfrak{C}(X)$, we have $\mathbb{F} \bullet \mathbb{F}_\emptyset = \mathbb{F}$ which implies by definition that $\mathbb{F}_\emptyset \leq \mathbb{F}$. Similarily, it is easy to see that for all $\mathbb{F} \in \mathfrak{C}(X)$, we have $\mathbb{F} \bullet (X, \emptyset) = (X, \emptyset)$ which implies by definition that $\mathbb{F} \leq (X, \emptyset)$.

Part 2: If X is an infinite set then $\mathfrak{C}_{\text{fin}}(X)$ is a sublattice of $\mathfrak{C}(X)$ with the same bottom and no top. It is clear that for every $\mathbb{F}_1 \in \mathfrak{C}_{\text{fin}}(X)$ and $\mathbb{F}_2 \in \mathfrak{C}(X)$ such that $\mathbb{F}_2 \leq \mathbb{F}_1$, we have that $\mathbb{F}_2 \in \mathfrak{C}_{\text{fin}}(X)$. It follows that $\mathfrak{C}_{\text{fin}}(X)$ is a sublattice of $\mathfrak{C}(X)$ with $\mathbb{F}_\emptyset$ as bottom. Moreover, if follows from the definition of $\bullet$ that if $\mathfrak{C}_{\text{fin}}(X)$ with X infinite would have a top $(\mathcal{F}, \mathcal{P})$, we would have $S \subseteq \mathcal{F}$ for all $S \subseteq_{\text{fin}} X$. This means that $\mathcal{F}$ should be equal to X, which is not possible.

Part 3: $\mathfrak{C}_{\text{eql}}(X)$ is a sublattice of $\mathfrak{C}(X)$ and it is a Boolean lattice with bottom $(X, 2^X)$, same top of $\mathfrak{C}(X)$, and complement $(\cdot)^\dagger$. It is clear that for every $\mathbb{F}_1, \mathbb{F}_2 \in \mathfrak{C}_{\text{eql}}(X)$ we have that $\mathbb{F}_1 \bullet \mathbb{F}_2 \in \mathfrak{C}_{\text{eql}}(X)$ and

$\mathbb{F}_1 \star \mathbb{F}_2 \in \mathfrak{C}_{\text{eql}}(X)$. It follows that $\mathfrak{C}_{\text{eql}}(X)$ is a sublattice of $\mathfrak{C}(X)$ with $(X, \emptyset)$ as top. Moreover, it is easy to see that $(X, 2^X) \in \mathfrak{C}_{\text{eql}}(X)$ and that for all $\mathbb{F}_1 \in \mathfrak{C}_{\text{eql}}(X)$, we have $(X, 2^X) \bullet \mathbb{F} = \mathbb{F}$. We prove the distributive law by using Lemmas 3.4 and 3.6: let $\mathbb{F}_1 = (X, \mathcal{P}_1), \mathbb{F}_2 = (X, \mathcal{P}_2), \mathbb{F}_3 = (X, \mathcal{P}_3) \in \mathfrak{C}_{\text{eql}}(X)$, so

$$\begin{aligned} \mathbb{F}_1 \star (\mathbb{F}_2 \bullet \mathbb{F}_3) &= (X, \mathcal{P}_1 \cup (\mathcal{P}_2 \cap \mathcal{P}_3)) \\ &= (X, (\mathcal{P}_1 \cup \mathcal{P}_2) \cap (\mathcal{P}_1 \cup \mathcal{P}_3)) = (\mathbb{F}_1 \star \mathbb{F}_2) \bullet (\mathbb{F}_1 \star \mathbb{F}_3) \end{aligned}$$

Finally, $\mathbb{F}_1 \star \mathbb{F}_1^{\dagger} = (X, 2^X)$ and $\mathbb{F}_1 \bullet \mathbb{F}_1^{\dagger} = (X, \emptyset)$ follow by Lemmas 3.4, 3.6. □

3.3. Lattices of AFMs

Notation 3.9. If v is a function and $X \subseteq \text{dom}(v)$ then $v|_X$ is the restriction of the function to the domain X.

Definition 3.10 (Sets of extensional AFMs). Let X be a set of features, let Y a set of attributes, let $\mathcal{D}$ be a set of domains, let $\ddot{\alpha}$ be a function of $Y \to X$ and let $\ddot{\delta}$ be a function of $Y \to \mathcal{D}$. We denote:

- $\mathfrak{A}(X, Y, \ddot{\alpha}, \mathcal{D}, \ddot{\delta})$ the set of all extensional AFMs $(\mathcal{F}, \mathcal{A}, \alpha, \mathcal{D}, \delta, \mathcal{V})$ such that $\mathcal{F} \subseteq X$, $\mathcal{A} \subseteq \ddot{\alpha}^{-1}(\mathcal{F})$, $\alpha = \ddot{\alpha}|_{\mathcal{A}}$ and $\delta = \ddot{\delta}|_{\mathcal{A}}$ (i.e. α, δ are the restrictions of $\ddot{\alpha}$ and $\ddot{\delta}$ to $\mathcal{A}$, respectively);
- $\mathfrak{A}_{\text{fin}}(X, Y, \ddot{\alpha}, \mathcal{D}, \ddot{\delta})$ the set of AFMs $(\mathcal{F}, \mathcal{A}, \alpha, \mathcal{D}, \delta, \mathcal{V}) \in \mathfrak{A}(X, Y, \ddot{\alpha}, \mathcal{D}, \ddot{\delta})$ such that $\mathcal{F} \subseteq_{\text{fin}} X$ and $\mathcal{A} \subseteq_{\text{fin}} \ddot{\alpha}^{-1}(\mathcal{F})$; and
- $\mathfrak{A}_{\text{eql}}(X, Y, \ddot{\alpha}, \mathcal{D}, \ddot{\delta})$ is the set of AFMs $(\mathcal{F}, \mathcal{A}, \alpha, \mathcal{D}, \delta, \mathcal{V}) \in \mathfrak{A}(X, Y, \ddot{\alpha}, \mathcal{D}, \ddot{\delta})$ such that $\mathcal{F} = X$, $\mathcal{A} = Y$, $\alpha = \ddot{\alpha}$ and $\delta = \ddot{\delta}$.

Clearly, $\mathfrak{A}_{\text{fin}}(X, Y, \ddot{\alpha}, \mathcal{D}, \ddot{\delta})$ and $\mathfrak{A}_{\text{eql}}(X, Y, \ddot{\alpha}, \mathcal{D}, \ddot{\delta})$ are subsets of $\mathfrak{A}(X, Y, \ddot{\alpha}, \mathcal{D}, \ddot{\delta})$. A further interesting set of extensional AFMs is $\mathfrak{A}_{\text{eql}}(X, Y, \ddot{\alpha}, \mathcal{D}, \ddot{\delta})$ where $X, Y, \mathcal{D}$ are finite sets: luckily, its main properties are that of the general case. It is worth to note that the AFMs in the above families are always compatible being restrictions of the same pattern.

Lemma 3.11 (Two criteria for the AFM fragment relation). *Let $\mathbb{A}_i$ be two AFMs ($i = 1, 2$) in $\mathfrak{A}(X, Y, \ddot{\alpha}, \mathcal{D}, \ddot{\delta})$. If $\mathbb{A}_i = (\mathcal{F}_i, \mathcal{A}_i, \alpha_i, \mathcal{D}, \delta_i, \mathcal{V}_i)$ then the following statements are equivalent:*

i) $\mathbb{A}_1 \leq \mathbb{A}_2$;

ii) $\mathbb{A}_1 \bullet \mathbb{A}_2 = \mathbb{A}_2$;

iii) $\mathcal{F}_1 \subseteq \mathcal{F}_2$, $\mathcal{A}_1 \subseteq \mathcal{A}_2$, $\alpha_1 \subseteq \alpha_2$, $\delta_1 \subseteq \delta_2$ *and* $\mathcal{V}_1 \supseteq \{(p_2|_{\mathcal{F}_1}, v_2|_{\mathcal{A}_1}) \mid (p_2, v_2) \in \mathcal{V}_2\}$.

Proof. *i) ⇒ ii)*. It is straightforward to check that $\mathbb{A} \bullet \mathbb{A} = \mathbb{A}$, for all $\mathbb{A}$. Then, by definition of $\leq$ (Definition 2.19) there is $\mathbb{A}' \in \mathfrak{A}(X, Y, \ddot{\alpha}, \mathcal{D}, \ddot{\delta})$ such that $\mathbb{A}_1 \bullet \mathbb{A}' = \mathbb{A}_2$. So, $\mathbb{A}_1 \bullet \mathbb{A}_2 = \mathbb{A}_1 \bullet (\mathbb{A}_1 \bullet \mathbb{A}') = (\mathbb{A}_1 \bullet \mathbb{A}_1) \bullet \mathbb{A}' = \mathbb{A}_1 \bullet \mathbb{A}' = \mathbb{A}_2$.

ii) ⇒ iii). Since the hypothesis $\mathbb{A}_1 \bullet \mathbb{A}_2 = \mathbb{A}_2$ and Definition 2.17, we have $\mathcal{F}_1 \subseteq \mathcal{F}_2$ $\mathcal{A}_1 \subseteq \mathcal{A}_2$, $\alpha_1 \subseteq \alpha_2$, $\delta_1 \subseteq \delta_2$ and $\mathcal{V}_2$ must be equal to

$$\{(p_1 \cup p_2, v_1 \oplus v_2) \mid (p_i, v_i) \in \mathcal{V}_i \text{ and } p_1 \cap \mathcal{F}_2 = p_2 \cap \mathcal{F}_1 \text{ and } v_1 \Subset v_2\}$$

which ensures that: for all $(p_2, v_2) \in \mathcal{V}_2$ there is (p_1, v_1) such that $p_1 = p_2 \cap \mathcal{F}_1$ and $v_1 \subseteq v_2$. More compactly, $\{(p_2|_{\mathcal{F}_1}, v_2|_{\mathcal{A}_1}) \mid (p_2, v_2) \in \mathcal{V}_2\} \subseteq \mathcal{V}_1$ must hold.

iii) ⇒ i). By hypothesis, it is clear that

$$(\mathcal{F}_1 \cup \mathcal{F}_2, \mathcal{A}_1 \cup \mathcal{A}_2, \alpha_1 \oplus \alpha_2, \mathcal{D}, \delta_1 \oplus \delta_2, \mathcal{V}) = (\mathcal{F}_2, \mathcal{A}_2, \alpha_2, \mathcal{D}, \delta_2, \mathcal{V}_2)$$

where $\mathcal{V} = \{(p_1 \cup p_2, v_1 \oplus v_2) \mid (p_i, v_i) \in \mathcal{V}_i,\ p_1 \cap \mathcal{F}_2 = p_2 \cap \mathcal{F}_1 \text{ and } v_1 \Subset v_2\})$, i.e. $\mathbb{A}_1 \bullet \mathbb{A}_2 = \mathbb{A}_2$. This implies, by definition of $\leq$, that $\mathbb{A}_1 \leq \mathbb{A}_2$. □

Lemma 3.12 (The operator $\bullet$ on $\mathfrak{A}_{\text{eql}}(X, Y, \ddot{\alpha}, \mathcal{D}, \ddot{\delta})$). *Let $\mathbb{A}_i \in \mathfrak{A}_{\text{eql}}(X, Y, \ddot{\alpha}, \mathcal{D}, \ddot{\delta})$ be two AFMs ($i = 1, 2$). If $\mathbb{A}_i = (X, Y, \ddot{\alpha}, \mathcal{D}, \ddot{\delta}, \mathcal{V}_i)$ then*

$$\mathbb{A}_1 \bullet \mathbb{A}_2 = (X, Y, \ddot{\alpha}, \mathcal{D}, \ddot{\delta}, \mathcal{V}_1 \cap \mathcal{V}_2).$$

Proof. $\{(p_1 \cup p_2, v_1 \oplus v_2) \mid (p_i, v_i) \in \mathcal{V}_i, p_1 \cap X = p_2 \cap X \text{ and } v_1 \Subset v_2\}$ is the set of AFM-products of $\mathbb{A}_1 \bullet \mathbb{A}_2$ in accord with Definition 2.17. Moreover, if $(p_i, v_i) \in \mathcal{V}_i$ then then $\text{dom}(v_i) = \ddot{\alpha}^{-1}(p_i)$, by Definition 2.6; hence, $(p_1, v_1) \in \mathcal{V}_1$, $(p_2, v_2) \in \mathcal{V}_2$ and $p_1 = p_2$ imply $\text{dom}(v_1) = \ddot{\alpha}^{-1}(p_1) = \ddot{\alpha}^{-1}(p_2) = \text{dom}(v_2)$. Therefore, $v_1 \Subset v_2$ means $v_1 = v_2$ and the proof is done. □

If γ_1, γ_2 are functions such that $\gamma_1 \Subset \gamma_2$ then, we denote $\gamma_1 \ominus \gamma_2$ is the restriction of the two functions to the common domain.

Definition 3.13 (Meet for AFMs). Let $\mathbb{A}_i$ be two AFMs in $\mathfrak{A}(X, Y, \ddot{\alpha}, \mathcal{D}, \ddot{\delta})$ such that $\mathbb{A}_i = (\mathcal{F}_i, \mathcal{A}_i, \alpha_i, \mathcal{D}, \delta_i, \mathcal{V}_i)$ where $i = 1, 2$. We define $\mathbb{A}_1 \star \mathbb{A}_2 = (\mathcal{F}_1 \cap \mathcal{F}_2, \mathcal{A}_1 \cap \mathcal{A}_2, \alpha_1 \ominus \alpha_2, \mathcal{D}, \delta_1 \ominus \delta_2, \mathcal{V})$ where $\mathcal{V} = \{(p_1|_{\mathcal{F}_2}, v_1|_{\mathcal{A}_2}) \mid (p_1, v_1) \in \mathcal{V}_1\} \cup \{(p_2|_{\mathcal{F}_1}, v_2|_{\mathcal{A}_1}) \mid (p_2, v_2) \in \mathcal{V}_2\}$.

Lemma 3.14 (The operator $\star$ on $\mathfrak{A}_{\text{eql}}(X, Y, \ddot{\alpha}, \mathcal{D}, \ddot{\delta})$). *Let $\mathbb{A}_i \in \mathfrak{A}_{\text{eql}}(X, Y, \ddot{\alpha}, \mathcal{D}, \ddot{\delta})$ be two AFMs ($i = 1, 2$). If $\mathbb{A}_i = (X, Y, \ddot{\alpha}, \mathcal{D}, \ddot{\delta}, \mathcal{V}_i)$ then*

$$\mathbb{A}_1 \star \mathbb{A}_2 = (X, Y, \ddot{\alpha}, \mathcal{D}, \ddot{\delta}, \mathcal{V}_1 \cup \mathcal{V}_2).$$

Proof. We have $\mathbb{A}_1 \star \mathbb{A}_2 = (\mathcal{F}_1 \cap \mathcal{F}_2, \mathcal{A}_1 \cap \mathcal{A}_2, \alpha_1 \ominus \alpha_2, \mathcal{D}, \delta_1 \ominus \delta_2, \mathcal{V})$ where $\mathcal{V} = \{(p_1|_X, v_1|_Y) \mid (p_1, v_1) \in \mathcal{V}_1\} \cup \{(p_2|_X, v_2|_Y) \mid (p_2, v_2) \in \mathcal{V}_2\}$ according to the definition of $\star$. The proof is easy, since $\mathcal{V} = \mathcal{V}_1 \cup \mathcal{V}_2$. □

Definition 3.15 (Complement on $\mathfrak{A}_{\text{eql}}(X, Y, \ddot{\alpha}, \mathcal{D}, \ddot{\delta})$). Let $\mathbb{A}$ be an AFM in $\mathfrak{A}(X, Y, \ddot{\alpha}, \mathcal{D}, \ddot{\delta})$ such that $\mathbb{A} = (\mathcal{F}, \mathcal{A}, \alpha, \mathcal{D}, \delta, \mathcal{V})$. We define:
$\mathbb{A}^{\dagger} = (\mathcal{F}, \mathcal{A}, \alpha, \mathcal{D}, \delta, \mathbf{all}(\mathcal{F}, \mathcal{A}, \alpha, \mathcal{D}, \delta) \setminus \mathcal{V})$.

Theorem 3.16 (Lattices of AFMs over a set of features).

1. *$(\mathfrak{A}(X, Y, \ddot{\alpha}, \mathcal{D}, \ddot{\delta}), \leq)$ is a bounded lattice with join $\bullet$, meet $\star$, with bottom* $\mathbf{trivial}(\emptyset, \emptyset, \bot, \mathcal{D}, \bot) = (\emptyset, \emptyset, \bot, \mathcal{D}, \bot, \{(\emptyset, \bot)\})$ *and top* $\mathbf{void}(X, Y, \ddot{\alpha}, \mathcal{D}, \ddot{\delta}) = (X, Y, \ddot{\alpha}, \mathcal{D}, \ddot{\delta}, \emptyset)$.

2. *If X is an infinite set then $\mathfrak{A}_{\text{fin}}(X, Y, \ddot{\alpha}, \mathcal{D}, \ddot{\delta})$ is a sublattice of $\mathfrak{A}(X, Y, \ddot{\alpha}, \mathcal{D}, \ddot{\delta})$ with the same bottom and no top.*

3. *$\mathfrak{A}_{\text{eql}}(X, Y, \ddot{\alpha}, \mathcal{D}, \ddot{\delta})$ is a sublattice of $\mathfrak{A}(X, Y, \ddot{\alpha}, \mathcal{D}, \ddot{\delta})$ with the same top; and, it forms a Boolean lattice with complement $(\cdot)^{\dagger}$ and bottom* $\mathbf{trivial}((X, Y, \ddot{\alpha}, \mathcal{D}, \ddot{\delta}) = (X, Y, \ddot{\alpha}, \mathcal{D}, \ddot{\delta}, \boldsymbol{all}(X, Y, \ddot{\alpha}, \mathcal{D}, \ddot{\delta}))$.

Proof. Let $\mathbb{A}_1 \leq \mathbb{A}_2 \leq \mathbb{A}_3$. The proof follows the same pattern of of Theorem 3.8. Let first prove that $\leq$ is a partial order.

- *Reflexivity.* In $\mathfrak{A}(X)$ and $\mathfrak{A}_{\text{fin}}(X)$ holds, because by Definition 2.17,

$$(\mathcal{F}, \mathcal{A}, \alpha, \mathcal{D}, \delta, \mathcal{V}) \bullet (\emptyset, \emptyset, \bot, \mathcal{D}, \bot, \{(\emptyset, \emptyset)\}) = (\mathcal{F}, \mathcal{A}, \alpha, \mathcal{D}, \delta, \mathcal{V})$$

  because, for all $(p, v) \in \mathcal{V}$ we have $p \cap \emptyset = \emptyset \cap \mathcal{F}$ and $\bot \not\simeq v$. In $\mathfrak{A}_{\text{eql}}(X)$, we have that each AFM has shape $((X, \mathcal{P}), Y, \ddot{\alpha}, \mathcal{D}, \ddot{\delta}, \mathcal{V})$, thus

$$((X, \mathcal{P}), Y, \ddot{\alpha}, \mathcal{D}, \ddot{\delta}, \mathcal{V}) \bullet ((X, 2^X), Y, \ddot{\alpha}, \mathcal{D}, \ddot{\delta}, \mathbb{V}) = ((X, \mathcal{P}), Y, \ddot{\alpha}, \mathcal{D}, \ddot{\delta}, \mathcal{V})$$

  where $\mathbb{V}$ is the set of all possible AFM-products, by using Lemma 5.10.

- *Antisymmetry.* The proof is the same for $\mathfrak{A}(X)$, $\mathfrak{A}_{\text{fin}}(X)$, $\mathfrak{A}_{\text{eql}}(X)$: it follows by associativity, commutativity and idempotency of $\bullet$ for AFMs. If $\mathbb{A}_2 \leq \mathbb{A}_1$ and $\mathbb{A}_1 \leq \mathbb{A}_2$ then, by hypothesis there are $\mathbb{A}, \mathbb{A}'$ such that $\mathbb{A}_1 = \mathbb{A}_2 \bullet \mathbb{A}$ and $\mathbb{A}_2 = \mathbb{A}_1 \bullet \mathbb{A}'$; therefore,

$$\begin{aligned} \mathbb{A}_1 &= \mathbb{A}_2 \bullet \mathbb{A} = (\mathbb{A}_1 \bullet \mathbb{A}) \bullet \mathbb{A} = ((\mathbb{A}_2 \bullet \mathbb{A}) \bullet \mathbb{A}') \bullet \mathbb{A} \\ &= \mathbb{A}_2 \bullet (\mathbb{A} \bullet \mathbb{A}) \bullet \mathbb{A}' = (\mathbb{A}_2 \bullet \mathbb{A}) \bullet \mathbb{A}' = \mathbb{A}_1 \bullet \mathbb{A}' = \mathbb{A}_2 \,. \end{aligned}$$

- *Transitivity.* The proof is the same for $\mathfrak{A}(X)$, $\mathfrak{A}_{\text{fin}}(X)$, $\mathfrak{A}_{\text{eql}}(X)$. Let $\mathbb{A}, \mathbb{A}'$ such that $\mathbb{A}_3 = \mathbb{A}_2 \bullet \mathbb{A}$ and $\mathbb{A}_2 = \mathbb{A}_1 \bullet \mathbb{A}'$. Clearly, $\mathbb{A}_3 = \mathbb{A}_2 \bullet \mathbb{A} = (\mathbb{A}_1 \bullet \mathbb{A}') \bullet \mathbb{A} = \mathbb{A}_1 \bullet (\mathbb{A}' \bullet \mathbb{A})$ which ensures that $\mathbb{A}_1 \leq \mathbb{A}_3$.

Part 1: $(\mathfrak{A}(X, Y, \ddot{\alpha}, \mathcal{D}, \ddot{\delta}), \leq)$ is a bounded lattice with join $\bullet$, meet $\star$, whit bottom $(\emptyset, \emptyset, \bot, \mathcal{D}, \bot, \{(\emptyset, \emptyset)\})$ and top $(X, Y, \ddot{\alpha}, \mathcal{D}, \ddot{\delta}, \emptyset)$. Let $\uparrow \mathbb{A}$ be the set of upper bounds of $\mathbb{A}$ w.r.t. $\leq$, viz. $\{\mathbb{A}' \mid \mathbb{A} \leq \mathbb{A}'\}$; and, let $\downarrow \mathbb{A}$ be the set of lower bounds of $\mathbb{A}$ w.r.t. $\leq$, viz. $\{\mathbb{A}' \mid \mathbb{A}' \leq \mathbb{A}\}$.

- $\mathbb{A}_i \leq \mathbb{A}_1 \bullet \mathbb{A}_2$ $(i = 1, 2)$ by definition of $\leq$; so, $\mathbb{A}_1 \bullet \mathbb{A}_2 \in (\uparrow \mathbb{A}_1) \cap (\uparrow \mathbb{A}_2)$. Moreover, for all common upper bounds $\mathbb{A} \in (\uparrow \mathbb{A}_1) \cap (\uparrow \mathbb{A}_2)$, we have, by Lemma 3.11, $\mathbb{A} = \mathbb{A}_1 \bullet \mathbb{A} = \mathbb{A}_1 \bullet (\mathbb{A}_2 \bullet \mathbb{A}) = (\mathbb{A}_1 \bullet \mathbb{A}_2) \bullet \mathbb{A}$. Thus, we conclude that $\mathbb{A}_1 \bullet \mathbb{A}_2$ is the join.

- Let us assume $\mathbb{A} = (\mathcal{F}_1 \cap \mathcal{F}_2, \mathcal{A}_1 \cap \mathcal{A}_2, \alpha_1 \cap \alpha_2, \mathcal{D}, \delta_1 \cap \delta_2, \mathcal{V})$ where $\mathcal{V} = \{(p_1|_{\mathcal{F}_2}, v_1|_{\mathcal{A}_2}) \mid (p, v) \in \mathcal{V}_1\} \cup \{(p_2|_{\mathcal{F}_1}, v_2|_{\mathcal{A}_1}) \mid (p, v) \in \mathcal{V}_2\}$. By Lemma 3.11, it follows that $\mathbb{A} \in (\downarrow \mathbb{A}_1) \cap (\downarrow \mathbb{A}_2)$. Moreover, for all $\mathbb{A}' \in (\downarrow \mathbb{A}_1) \cap (\downarrow \mathbb{A}_2)$, it is easy to check that $\mathbb{A}' \leq \mathbb{A}$ by Lemma 3.11. Thus, $\star$ play the role of meet.

- Let $\mathbb{A}^{\perp} = (\emptyset, \emptyset, \perp, \mathcal{D}, \perp, \{(\emptyset, \emptyset)\})$. For all $\mathbb{A} \in \mathfrak{A}(X, Y, \ddot{\alpha}, \mathcal{D}, \ddot{\delta}), \leq)$, we have $\mathbb{A} \bullet \mathbb{A}^{\perp} = \mathbb{A}$ (c.f. reflexivity) which means $\mathbb{A}^{\perp} \leq \mathbb{A}$ by $\leq$-definition, viz. we found the bottom. Let $\mathbb{A} \in \mathfrak{A}(X, Y, \ddot{\alpha}, \mathcal{D}, \ddot{\delta})$ be $(\mathcal{F}, \mathcal{A}, \alpha, \mathcal{D}, \delta, \mathcal{V})$ then

$$(\mathcal{F}, \mathcal{A}, \alpha, \mathcal{D}, \delta, \mathcal{V}) \bullet (X, Y, \ddot{\alpha}, \mathcal{D}, \ddot{\delta}, \emptyset) = (X, Y, \ddot{\alpha}, \mathcal{D}, \ddot{\delta}, \emptyset)$$

  which means $\mathbb{A} \leq ((X, \emptyset), Y, \ddot{\alpha}, \mathcal{D}, \ddot{\delta}, \emptyset)$, for all $\mathbb{A}$, viz. we found the top.

Part 2: If X is an infinite set then $\mathfrak{A}_{\text{fin}}(X, Y, \ddot{\alpha}, \mathcal{D}, \ddot{\delta})$ is a sublattice of $\mathfrak{A}(X, Y, \ddot{\alpha}, \mathcal{D}, \ddot{\delta})$ with the same bottom and no top. It is clear that for every $\mathbb{A}_1 \in \mathfrak{A}_{\text{fin}}(X)$ and $\mathbb{A}_2 \in \mathfrak{A}(X)$ such that $\mathbb{A}_2 \leq \mathbb{A}_1$, we have that $\mathbb{A}_2 \in \mathfrak{A}_{\text{fin}}(X)$. It follows that $\mathfrak{A}_{\text{fin}}(X)$ is a sublattice of $\mathfrak{A}(X)$ with $\mathbb{A}_{\perp}$ as bottom. Moreover, if follows from the definition of $\bullet$ that if $\mathfrak{A}_{\text{fin}}(X)$ with X infinite would have a top $(\mathcal{F}^{\top}, \mathcal{A}^{\top}, \alpha^{\top}, \mathcal{D}, \delta^{\top}, \mathcal{V}^{\top})$. But, for all AFMs with S as set of feature, we would that $S \subseteq \mathcal{F}^{\top}$ and, thence, $S \subseteq_{\text{fin}} X$. This means that $\mathcal{F}$ should be equal to X, which is not possible. (Similar reasoning holds for Y or $\mathcal{D}$ infinite).

Part 3: $\mathfrak{A}_{\text{eql}}(X, Y, \ddot{\alpha}, \mathcal{D}, \ddot{\delta})$ is a sublattice of $\mathfrak{A}(X, Y, \ddot{\alpha}, \mathcal{D}, \ddot{\delta})$ with the same top; and, it forms a Boolean lattice with complement $(\cdot)^{\dagger}$ and bottom $(X, Y, \ddot{\alpha}, \mathcal{D}, \ddot{\delta}, \mathbf{all}(X, Y, \ddot{\alpha}, \mathcal{D}, \ddot{\delta}))$. Clearly, if $\mathbb{A}_1, \mathbb{A}_2 \in \mathfrak{A}_{\text{eql}}(X, Y, \ddot{\alpha}, \mathcal{D}, \ddot{\delta})$ then also $((X, \emptyset), Y, \ddot{\alpha}, \mathcal{D}, \ddot{\delta}, \emptyset)$, $\mathbb{A}_1 \bullet \mathbb{A}_2, \mathbb{A}_1 \star \mathbb{A}_2 \in \mathfrak{A}_{\text{eql}}(X, Y, \ddot{\alpha}, \mathcal{D}, \ddot{\delta})$. Therefore $\mathfrak{A}_{\text{eql}}(X, Y, \ddot{\alpha}, \mathcal{D}, \ddot{\delta})$ is a sublattice of $\mathfrak{A}(X, Y, \ddot{\alpha}, \mathcal{D}, \ddot{\delta})$ with $((X, \emptyset), Y, \ddot{\alpha}, \mathcal{D}, \ddot{\delta}, \emptyset)$ as top. Furthermore, if $\mathbb{A} = (X, Y, \ddot{\alpha}, \mathcal{D}, \ddot{\delta}, \mathcal{V}) \in \mathfrak{A}_{\text{eql}}(X, Y, \ddot{\alpha}, \mathcal{D}, \ddot{\delta})$ then $(X, Y, \ddot{\alpha}, \mathcal{D}, \ddot{\delta}, \mathbf{all}(X, Y, \ddot{\alpha}, \mathcal{D}, \ddot{\delta})) \bullet \mathbb{A} = \mathbb{A}$ by Lemma 5.10. We prove the distributive law by using Lemmas 5.10 and 3.14: let $(X, Y, \ddot{\alpha}, \mathcal{D}, \ddot{\delta}, \mathcal{V}_i) \in \mathfrak{C}_{\text{eql}}(X)$ where $i = 1, 2, 3$, we have:

$$\begin{aligned}\mathbb{A}_1 \star (\mathbb{A}_2 \bullet \mathbb{A}_3) &= (X, Y, \ddot{\alpha}, \mathcal{D}, \ddot{\delta}, \mathcal{P}_1 \cup (\mathcal{P}_2 \cap \mathcal{P}_3)) \\ &= (X, Y, \ddot{\alpha}, \mathcal{D}, \ddot{\delta}, (\mathcal{P}_1 \cup \mathcal{P}_2) \cap (\mathcal{P}_1 \cup \mathcal{P}_3)) = (\mathbb{A}_1 \star \mathbb{A}_2) \bullet (\mathbb{A}_1 \star \mathbb{A}_3)\end{aligned}$$

Finally, it is easy to check that $\mathbb{A}_1 \star \mathbb{A}_1^{\dagger} = (X, Y, \ddot{\alpha}, \mathcal{D}, \ddot{\delta}, \mathbf{all}(X, Y, \ddot{\alpha}, \mathcal{D}, \ddot{\delta}))$ and $\mathbb{A}_1 \bullet \mathbb{A}_1^{\dagger} = (X, Y, \ddot{\alpha}, \mathcal{D}, \ddot{\delta}, \emptyset)$ by using Lemmas 5.10, 3.14. □

3.4. On FM Fragments and Interfaces

FM slices were defined by Acher et al. [36] as a unary operator Π_Y that restricts a FM to the set Y of features.

Definition 3.17 (FM slice operator). Let $\mathbb{F} = (\mathcal{F}, \mathcal{P})$ be a FM. The *slice* operator Π_Y on FMs, where Y is a set of features, is defined by: $\Pi_Y(\mathbb{F}) = (\mathcal{F} \cap Y, \mathcal{P}|_Y)$.

More recently, Schröter et al. [18] introduced the following notion of FM interface.

Definition 3.18 (FM interface relation). A feature model $\mathbb{F}_1 = (\mathcal{F}_1, \mathcal{P}_1)$ is an *interface* of FM $\mathbb{F}_2 = (\mathcal{F}_2, \mathcal{P}_2)$, denoted as $\mathbb{F}_1 \preceq \mathbb{F}_2$, whenever both $\mathcal{F}_1 \subseteq \mathcal{F}_2$ and $\mathcal{P}_1 = \mathcal{P}_2|_{\mathcal{F}_1}$ hold.

Remark 3.19 (On FM interfaces and slices). As pointed out in [18], FM slices and interfaces are closely related. Namely: $\mathbb{F}_1 \preceq \mathbb{F}_2$ holds if and only if there exists a set of features Y such that $\mathbb{F}_1 = \Pi_Y(\mathbb{F}_2)$.

Example 3.20 (A slice of the glibc FM). Applying the operator $\Pi_{\{\mathsf{glibc},\ \mathsf{glibc{:}v}\}}$ to the FM $\mathbb{F}_{\mathsf{glibc}}$ of Example 2.4 yields the FM

$$\mathcal{F} = \{\mathsf{glibc},\ \mathsf{glibc{:}v}\} \qquad\qquad \mathcal{P} = \{\emptyset, \{\mathsf{glibc}\}, \{\mathsf{glibc},\ \mathsf{glibc{:}v}\}\},$$

which (according to Remark 3.19) is an interface for $\mathbb{F}_{\mathsf{glibc}}$.

The following theorem points out the relationship between the FM interface relation (designed to abstract away a set of features from a FM) and the FM fragment relation (designed to support FM decomposition).

Theorem 3.21 (Interfaces are fragments). *If* $\mathbb{F}_1 \preceq \mathbb{F}_2$ *then* $\mathbb{F}_1 \leq \mathbb{F}_2$.

Proof. Immediate by Definition 3.18 and Lemma 3.3. □

We conclude this section by providing some algebraic properties that relate the slice operator and the interface an fragment relations.

Lemma 3.22 (Monotonocity properties of the FM slice operator). *For all* $\mathcal{F}, \mathcal{F}_1, \mathcal{F}_2 \subseteq X$ *and* $\mathbb{F}, \mathbb{F}_1, \mathbb{F}_2 \in \mathfrak{C}(X)$

1. *If* $\mathcal{F}_1 \subseteq \mathcal{F}_2$ *then* $\Pi_{\mathcal{F}_1}(\mathbb{F}) \preceq \Pi_{\mathcal{F}_2}(\mathbb{F})$.
2. *If* $\mathcal{F}_1 \subseteq \mathcal{F}_2$ *then* $\Pi_{\mathcal{F}_1}(\mathbb{F}) \leq \Pi_{\mathcal{F}_2}(\mathbb{F})$.
3. *If* $\mathbb{F}_1 \preceq \mathbb{F}_2$ *then* $\Pi_{\mathcal{F}}(\mathbb{F}_1) \preceq \Pi_{\mathcal{F}}(\mathbb{F}_2)$.
4. *If* $\mathbb{F}_1 \leq \mathbb{F}_2$ *then* $\Pi_{\mathcal{F}}(\mathbb{F}_1) \leq \Pi_{\mathcal{F}}(\mathbb{F}_2)$.

Proof.

1. Let $\mathbb{F} = (\mathcal{F}, \mathcal{P})$. Since $\mathcal{F}_1 \subseteq \mathcal{F}_2$, we have $\mathcal{F}_1 \cap \mathcal{F} \subseteq \mathcal{F}_2 \cap \mathcal{F}$ and $\mathcal{P}|_{\mathcal{F}_1} = \mathcal{P}|_{\mathcal{F}_1 \cap \mathcal{F}_2} = \mathcal{P}|_{\mathcal{F}_2}|_{\mathcal{F}_1}$, so we conclude by Definition 3.18.
2. Immediate by Lemma 3.22.1 and Theorem 3.21.
3. By Definition 3.18, we have that $\mathcal{F}_1 \subseteq \mathcal{F}_2$ and $\mathcal{P}_1 = \mathcal{P}_2|_{\mathcal{F}_1}$. Consequently, for all $\mathcal{F} \subseteq X$, we have $(\mathcal{F}_1 \cap \mathcal{F}) \subseteq (\mathcal{F}_2 \cap \mathcal{F})$ and $\mathcal{P}_1|_{\mathcal{F}} = \mathcal{P}_2|_{\mathcal{F}_1}|_{\mathcal{F}}$. Hence, $\Pi_{\mathcal{F}}(\mathbb{F}_1) \preceq \Pi_{\mathcal{F}}(\mathbb{F}_2)$ by Definition 3.18.
4. By Lemma 3.3, we have that $\mathcal{F}_1 \subseteq \mathcal{F}_2$ and $\mathcal{P}_1 \supseteq \mathcal{P}_2|_{\mathcal{F}_1}$. Consequently, for all $\mathcal{F} \subseteq X$, we have $(\mathcal{F}_1 \cap \mathcal{F}) \subseteq (\mathcal{F}_2 \cap \mathcal{F})$ and $\mathcal{P}_1|_{\mathcal{F}} \supseteq \mathcal{P}_2|_{\mathcal{F}_1}|_{\mathcal{F}}$. Hence, $\Pi_{\mathcal{F}}(\mathbb{F}_1) \leq \Pi_{\mathcal{F}}(\mathbb{F}_2)$ by Lemma 3.3.

We remark that Lemma 3.22.3 and Theorem 3.21 do not imply Lemma 3.22.4.

Theorem 3.23 (Algebraic properties of the FM slice operator). *For all* $\mathbb{F}_1, \mathbb{F}_2, \mathbb{F}_3 \in \mathfrak{C}(X)$ *and* $\mathcal{F}_4, \mathcal{F}_5 \subseteq X$*, we have*

$\leq$-Monotonicity. *If* $\mathbb{F}_1 \leq \mathbb{F}_2$ *and* $\mathcal{F}_4 \subseteq \mathcal{F}_5$*, then* $\Pi_{\mathcal{F}_4}(\mathbb{F}_1) \leq \Pi_{\mathcal{F}_5}(\mathbb{F}_2)$.

$\preceq$-Monotonicity. *If* $\mathbb{F}_1 \preceq \mathbb{F}_2$ *and* $\mathcal{F}_4 \subseteq \mathcal{F}_5$*, then* $\Pi_{\mathcal{F}_4}(\mathbb{F}_1) \preceq \Pi_{\mathcal{F}_5}(\mathbb{F}_2)$.

Commutativity. $\Pi_{\mathcal{F}_4}(\Pi_{\mathcal{F}_5}(\mathbb{F}_3)) = \Pi_{\mathcal{F}_5}(\Pi_{\mathcal{F}_4}(\mathbb{F}_3))$.

Proof. **$\leq$-Monotonicity.** Straightforward by Lemma 3.22.2 and Lemma 3.22.4.

$\preceq$-Monotonicity. Straightforward by Lemma 3.22.1 and Lemma 3.22.3.

Commutativity. In accordance with Definition 3.18, it is sufficient to observe that $\Pi_{\mathcal{F}_4}(\Pi_{\mathcal{F}_5}(\mathbb{F}_3)) = \Pi_{\mathcal{F}_4 \cup \mathcal{F}_5}(\mathbb{F}_3) = \Pi_{\mathcal{F}_5}(\Pi_{\mathcal{F}_4}(\mathbb{F}_3))$ holds. □

3.5. On AFM Fragments and Interfaces

We propose to extend the FM slice operator (cf. Definition 3.17) to AFMs by considering also the set of attributes that should be kept by the slicing.

Definition 3.24 (AFM slice operator). Let $\mathbb{A} = (\mathcal{F}, \mathcal{A}, \alpha, \mathcal{D}, \delta, \mathcal{V})$ be an AFM in $\mathfrak{A}(X, Y, \ddot{\alpha}, \mathcal{D}, \ddot{\delta})$, $\mathcal{F}' \subseteq X$ and $\mathcal{A}' \subseteq Y$. The *slice* operator $\Pi_{\mathcal{F}', \mathcal{A}'}$ on AFMs is defined by: $\Pi_{\mathcal{F}', \mathcal{A}'}(\mathbb{A}) = (\mathcal{F} \cap \mathcal{F}', \mathcal{A} \cap \mathcal{A}_R, \alpha|_{\mathcal{A}_R}, \mathcal{D}, \delta|_{\mathcal{A}_R}, \mathcal{V}|_{\mathcal{F}', \mathcal{A}'})$ such that $\mathcal{V}|_{\mathcal{F}', \mathcal{A}'} = \{(p|_{\mathcal{F}'}, v|_{\mathcal{A}_R}) \mid (p, v) \in \mathcal{V}\}$ where $\mathcal{A}_R = \alpha^{-1}(\mathcal{F}') \cap \mathcal{A}'$.

Note that if $\mathbb{A} = (\mathcal{F}, \mathcal{A}, \alpha, \mathcal{D}, \delta, \mathcal{V})$ is an AFM then $\alpha^{-1}(\mathcal{F}) = \mathcal{A}$ by Definition 2.6. Therefore, in the the above definition, it is natural to expect that $\mathcal{A}' \subseteq \alpha^{-1}(\mathcal{F}')$, albeit we do not ask it. We also extend the notion of FM interfaces (cf. Definition 3.18) to AFMs.

Definition 3.25 (AFM interface relation). Let $\mathbb{A}_i = (\mathcal{F}_i, \mathcal{A}_i, \alpha_i, \mathcal{D}, \delta_i, \mathcal{V})$ be two AFMs where $i = 1, 2$. $\mathbb{A}_1$ is an *interface* of $\mathbb{A}_2$ denoted as $\mathbb{A}_1 \preceq \mathbb{A}_2$, whenever $\mathcal{F}_1 \subseteq \mathcal{F}_2$, $\mathcal{A}_1 \subseteq \mathcal{A}_2$, $\alpha_1 \subseteq \alpha_2$, $\delta_1 \subseteq \delta_2$, $\mathcal{V}_1 = \{(p_2|_{\mathcal{F}_1}, v_2|_{\mathcal{A}_1}) \mid (p_2, v_2) \in \mathcal{V}_2\}$.

AFM slices and interfaces are closely related: $\mathbb{A}_1 \preceq \mathbb{A}_2$ holds if and only if $\mathbb{F}_1 = \Pi_{\mathcal{F}_1, \mathcal{A}_1}(\mathbb{F}_2)$ holds.

Theorem 3.26 (AFM interfaces are AFM fragments). *Let* $\mathbb{A}_i$ *be two AFMs where* $i = 1, 2$*. If* $\mathbb{A}_1 \preceq \mathbb{A}_2$ *then* $\mathbb{A}_1 \leq \mathbb{A}_2$.

Proof. Immediate by Definition 3.25 and Lemma 3.11. □

In what follows we present some algebraic properties that relate slices and fragments.

Lemma 3.27 (Monotonocity properties of AFM slice operator). *Let us assume* $\mathbb{A}, \mathbb{A}_1, \mathbb{A}_2 \in \mathfrak{A}(X, Y, \ddot{\alpha}, \mathcal{D}, \ddot{\delta})$*,* $\mathcal{F}, \mathcal{F}_1, \mathcal{F}_2 \subseteq X$ *and* $\mathcal{A}, \mathcal{A}_1, \mathcal{A}_2 \subseteq Y$ *.*

1. *If* $\mathcal{F}_1 \subseteq \mathcal{F}_2$ *and* $\mathcal{A}_1 \subseteq \mathcal{A}_2$ *then* $\Pi_{\mathcal{F}_1, \mathcal{A}_1}(\mathbb{A}) \preceq \Pi_{\mathcal{F}_2, \mathcal{A}_2}(\mathbb{A})$.

2. *If* $\mathcal{F}_1 \subseteq \mathcal{F}_2$ *and* $\mathcal{A}_1 \subseteq \mathcal{A}_2$ *then* $\Pi_{\mathcal{F}_1,\mathcal{A}_1}(\mathbb{A}) \leq \Pi_{\mathcal{F}_2,\mathcal{A}_2}(\mathbb{A})$.

3. *If* $\mathbb{A}_1 \preceq \mathbb{A}_2$ *then* $\Pi_{\mathcal{F},\mathcal{A}}(\mathbb{A}_1) \preceq \Pi_{\mathcal{F},\mathcal{A}}(\mathbb{A}_2)$.

4. *If* $\mathbb{A}_1 \leq \mathbb{A}_2$ *then* $\Pi_{\mathcal{F},\mathcal{A}}(\mathbb{A}_1) \leq \Pi_{\mathcal{F},\mathcal{A}}(\mathbb{A}_2)$.

Proof. 1. Let $\mathbb{A} = (\mathcal{F}, \mathcal{A}, \alpha, \mathcal{D}, \delta, \mathcal{V})$. In accord with Definition 3.24, $\Pi_{\mathcal{F}_i,\mathcal{A}_i}(\mathbb{A}) = (\mathcal{F}\cap\mathcal{F}_i, \mathcal{A}\cap\mathcal{A}_{R_i}, \alpha|_{\mathcal{A}_{R_i}}, \mathcal{D}, \delta|_{\mathcal{A}_{R_i}}, \mathcal{V}|_{\mathcal{F}_i,\mathcal{A}_i})$ such that $\mathcal{V}|_{\mathcal{F}_i,\mathcal{A}_i} = \{(p|_{\mathcal{F}_i}, v|_{\mathcal{A}_{R_i}}) \mid (p,v) \in \mathcal{V}\}$ where $\mathcal{A}_{R_i} = \alpha^{-1}(\mathcal{F}_i) \cap \mathcal{A}_i$ $(i = 1,2)$. Since $\mathcal{F}_1 \subseteq \mathcal{F}_2$ and $\mathcal{A}_1 \subseteq \mathcal{A}_2$, we have $\mathcal{F}\cap\mathcal{F}_1 \subseteq \mathcal{F}\cap\mathcal{F}_2$, $\mathcal{A}_{R_1} \subseteq \mathcal{A}_{R_1}$, $\mathcal{A}\cap\mathcal{A}_{R_1} \subseteq \mathcal{A}\cap\mathcal{A}_{R_2}$, $\alpha|_{\mathcal{A}_{R_1}} \subseteq \alpha|_{\mathcal{A}_{R_2}}$ and $\mathcal{V}|_{\mathcal{F}_1,\mathcal{A}_1} = \mathcal{V}|_{\mathcal{F}_1\cap\mathcal{F}_2,\mathcal{A}_1\cap\mathcal{A}_2} = (\mathcal{V}|_{\mathcal{F}_2,\mathcal{A}_2})|_{\mathcal{F}_1,\mathcal{A}_1}$. Thus the proof follows by Definition 3.25.

2. Immediate by Lemma 3.27.1 and Theorem 3.26.

3. Let $\mathbb{A}_i = (\mathcal{F}_i, \mathcal{A}_i, \alpha_i, \mathcal{D}, \delta_i, \mathcal{V}_i)$ where $i = 1,2$. By Definition 3.25, we have that $\mathcal{F}_1 \subseteq \mathcal{F}_2$, $\mathcal{A}_1 \subseteq \mathcal{A}_2$, $\alpha_1 \subseteq \alpha_2$, $\delta_1 \subseteq \delta_2$, $\mathcal{V}_1 = \{(p_2|_{\mathcal{F}_1}, v_2|_{\mathcal{A}_1}) \mid (p_2, v_2) \in \mathcal{V}_2\}$. Therefore, if $\mathcal{A}^1_R = \mathcal{A}_1 \cap \mathcal{A}$ and $\mathcal{A}^2_R = \mathcal{A}_2 \cap \mathcal{A}$ then $\mathcal{A}^2_R \subseteq \mathcal{A}^2_R$, $\mathcal{F}_1 \cap \mathcal{F} \subseteq \mathcal{F}_2 \cap \mathcal{F}$, $\mathcal{A}_1 \cap \mathcal{A}^1_R \subseteq \mathcal{A}_2 \cap \mathcal{A}^1_R$, $\alpha_1|_{\mathcal{A}^1_R} \subseteq \alpha_2|_{\mathcal{A}^2_R}$, $\delta_1|_{\mathcal{A}^1_R} \subseteq \delta_2|_{\mathcal{A}^1_R}$ and $\{(p_1|_{\mathcal{F}}, v_1|_{\mathcal{A}^R_1}) \mid (p_1, v_1) \in \mathcal{V}_1\} = \{(p_2|_{\mathcal{F}}|_{\mathcal{F}_1}, v_2|_{\mathcal{A}^2_R}|_{\mathcal{A}^1_R}) \mid (p_2, v_2) \in \mathcal{V}_2\}$. Thus the proof is done, by Definition 3.25.

4. Let $\mathbb{A}_i = (\mathcal{F}_i, \mathcal{A}_i, \alpha_i, \mathcal{D}, \delta_i, \mathcal{V}_i)$ where $i = 1,2$. By Lemma 3.11, we have that $\mathcal{F}_1 \subseteq \mathcal{F}_2$, $\mathcal{A}_1 \subseteq \mathcal{A}_2$, $\alpha_1 \subseteq \alpha_2$, $\delta_1 \subseteq \delta_2$, $\mathcal{V}_1 \supseteq \{(p_2|_{\mathcal{F}_1}, v_2|_{\mathcal{A}_1}) \mid (p_2, v_2) \in \mathcal{V}_2\}$. Therefore, if $\mathcal{A}^1_R = \mathcal{A}_1 \cap \mathcal{A}$ and $\mathcal{A}^2_R = \mathcal{A}_2 \cap \mathcal{A}$ then $\mathcal{A}^2_R \subseteq \mathcal{A}^2_R$, $\mathcal{F}_1 \cap \mathcal{F} \subseteq \mathcal{F}_2 \cap \mathcal{F}$, $\mathcal{A}_1 \cap \mathcal{A}^1_R \subseteq \mathcal{A}_2 \cap \mathcal{A}^1_R$, $\alpha_1|_{\mathcal{A}^1_R} \subseteq \alpha_2|_{\mathcal{A}^2_R}$, $\delta_1|_{\mathcal{A}^1_R} \subseteq \delta_2|_{\mathcal{A}^1_R}$ and $\{(p_1|_{\mathcal{F}}, v_1|_{\mathcal{A}^R_1}) \mid (p_1, v_1) \in \mathcal{V}_1\} \supseteq \{(p_2|_{\mathcal{F}}|_{\mathcal{F}_1}, v_2|_{\mathcal{A}^2_R}|_{\mathcal{A}^1_R}) \mid (p_2, v_2) \in \mathcal{V}_2\}$. Thus the proof is done, by Lemma 3.11.

We remark that Lemma 3.27.3 and Theorem 3.26 do not imply Lemma 3.27.4.

Theorem 3.28 (Algebraic properties of the AFM slice operator). *Let us assume* $\mathbb{A}_i \in \mathfrak{A}(X, Y, \ddot{\alpha}, \mathcal{D}, \ddot{\delta})$ *where* $i = 1,2,3$, $\mathcal{F}_4, \mathcal{F}_5 \subseteq X$ *and* $\mathcal{A}_6, \mathcal{A}_7 \subseteq Y$.

$\leq$-Monotonicity. *If* $\mathbb{A}_1 \leq \mathbb{A}_2$, $\mathcal{F}_4 \subseteq \mathcal{F}_5$, $\mathcal{A}_5 \subseteq \mathcal{A}_6$ *then* $\Pi_{\mathcal{F}_4,\mathcal{A}_6}(\mathbb{A}_1) \leq \Pi_{\mathcal{F}_5,\mathcal{A}_7}(\mathbb{A}_2)$.

$\preceq$-Monotonicity. *If* $\mathbb{A}_1 \leq \mathbb{A}_2$, $\mathcal{F}_4 \subseteq \mathcal{F}_5$, $\mathcal{A}_5 \subseteq \mathcal{A}_6$ *then* $\Pi_{\mathcal{F}_4,\mathcal{A}_6}(\mathbb{A}_1) \preceq \Pi_{\mathcal{F}_5,\mathcal{A}_7}(\mathbb{A}_2)$.

Commutativity. $\Pi_{\mathcal{F}_4,\mathcal{A}_6}(\Pi_{\mathcal{F}_5\mathcal{A}_7}(\mathbb{A}_3)) = \Pi_{\mathcal{F}_5,\mathcal{A}_7}(\Pi_{\mathcal{F}_4,\mathcal{A}_6}(\mathbb{A}_3))$.

Proof. **$\leq$-Monotonicity.** Straightforward by Lemma 3.27.2 and Lemma 3.27.4.

$\preceq$-Monotonicity. Straightforward by Lemma 3.27.1 and Lemma 3.27.3.

Commutativity. In accordance with Definition 3.25, it is sufficient to observe that $\Pi_{\mathcal{F}_4,\mathcal{A}_6}(\Pi_{\mathcal{F}_5,\mathcal{A}_7}(\mathbb{A}_3)) = \Pi_{\mathcal{F}_4\cap\mathcal{F}_5,\mathcal{A}_5\cap\mathcal{A}_7}(\mathbb{A}_3) = \Pi_{\mathcal{F}_5,\mathcal{A}_7}(\Pi_{\mathcal{F}_4,\mathcal{A}_6}(\mathbb{A}_3))$. $\square$

4. Logical Characterization of FM Operations and Relations

In Section 4.1 we introduce a mapping that associates each logical FM to its corresponding extensional representation (cf. Sect. 2.1). Then, in Section 4.2, we provide a logical characterization for the fragment relation ($\leq$), for the composition ($\bullet$) and the meet ($\star$) operations; for the the bottom of the Boolean lattice $\mathfrak{E}_{\text{eql}}(X)$ (the FM $\textbf{trivial}(X) = (X, 2^X)$), for the bottom of the bounded lattice $\mathfrak{E}(X)$ (the FM $\textbf{trivial}(\emptyset) = (\emptyset, \{\emptyset\})$) and for the top of the bounded lattice $\mathfrak{E}(X)$) (the FM $\textbf{void}(X) = (X, \emptyset)$); and for the complement operation ($(\cdot)^\dagger$). Finally, in Section 4.3, we provide a logical characterization for the slice operator (Π_Y) and for the interface relation ($\preceq$).

4.1. Relating Extensional and Logical FMs

As stated at the beginning of Sect. 3.2, in our theoretical development we consider also FMs with infinitely many features and products, where each product may have infinitely many features. The following definition introduces a notion for three different sets of logical FMs (see Definition 2.9) over a set of features (cf. Definition 3.10).

Definition 4.1 (Sets of logical FMs over a set of features).
Let X be a set of features. We denote:

- $\mathfrak{P}(X)$ the set of the logical FMs $(\mathcal{F}, \phi)$ such that $\mathcal{F} \subseteq X$;
- $\mathfrak{P}_{\text{fin}}(X)$ the subset of the finite elements of $\mathfrak{P}(X)$, i.e., $(\mathcal{F}, \phi)$ such that $\mathcal{F} \subseteq_{\text{fin}} X$; and
- $\mathfrak{P}_{\text{eql}}(X)$ the subset of elements of $\mathfrak{P}(X)$ that have exactly the features X, i.e., $(\mathcal{F}, \phi)$ such that $\mathcal{F} = X$.

We denote by $\text{ftrs}(\phi)$ the (finite) set of features occurring in a propositional formula ϕ, and as usual we say that ϕ is *ground* whenever $\text{ftrs}(\phi)$ is empty. We recall that an *interpretation* (a.k.a. truth assignment or valuation) $\mathcal{I}$ is a function which maps propositional variables to true or false [1, 37].

Definition 4.2 (Interpretation for propositions). Let $(\mathcal{F}, \mathcal{P})$ be an extensional FM and $p \in \mathcal{P}$. The interpretation that *represents* the product p is the function $\mathcal{I}_p^{\mathcal{F}} : \mathcal{F} \to \{\mathsf{true}, \mathsf{false}\}$ such that: $\mathcal{I}_p^{\mathcal{F}}(x) = \mathit{true}$ if $x \in p$, and $\mathcal{I}_p^{\mathcal{F}}(x) = \mathit{false}$ if $x \in \mathcal{F} \setminus p$.

As usual, $\text{dom}(\mathcal{I})$ denotes the domain of an interpretation $\mathcal{I}$ and we write $\mathcal{I} \models \phi$ to mean that the propositional formula ϕ is true under the interpretation $\mathcal{I}$ (i.e., $\text{ftrs}(\phi) \subseteq \text{dom}(\mathcal{I})$ and the ground formula obtained from ϕ by replacing each feature x occurring in ϕ by $\mathcal{I}(x)$ evaluates to true). We write $\models \phi$ to mean that ϕ is valid (i.e., it evaluates to true under all the interpretations $\mathcal{I}$ such that $\text{ftrs}(\phi) \subseteq \text{dom}(\mathcal{I})$). We write $\phi_1 \models \phi_2$ to mean that ϕ_2 is a logical consequence of ϕ_1 (i.e., for all interpretations $\mathcal{I}$ with $\text{ftrs}(\phi_1) \cup \text{ftrs}(\phi_2) \subseteq \text{dom}(\mathcal{I})$, if $\mathcal{I} \models \phi_1$ then

$\mathcal{I} \models \phi_2$), and we write $\phi_1 \equiv \phi_2$ to mean that ϕ_1 and ϕ_2 are logically equivalent (i.e., they are satisfied by exactly the same interpretations with domain including $\mathrm{ftrs}(\phi_1) \cup \mathrm{ftrs}(\phi_2)$). We recall that: (i) $\mathcal{I}_1$ is *included* in $\mathcal{I}_2$, denoted $\mathcal{I}_1 \subseteq \mathcal{I}_2$, whenever $\mathrm{dom}(\mathcal{I}_1) \subseteq \mathrm{dom}(\mathcal{I}_2)$ and $\mathcal{I}_1(x) = \mathcal{I}_2(x)$, for all $x \in \mathrm{dom}(\mathcal{I}_1)$; (ii) $\mathcal{I}_1$ and $\mathcal{I}_2$ are *compatible* whenever $\mathcal{I}_1(x) = \mathcal{I}_2(x)$, for all $x \in \mathrm{dom}(\mathcal{I}_1) \cap \mathrm{dom}(\mathcal{I}_2)$; and (iii) if $\mathcal{I}_1 \models \phi$ then its *restriction* $\mathcal{I}_0$ to $\mathrm{ftrs}(\phi)$ is such that $\mathcal{I}_0 \models \phi$ and, for all interpretations $\mathcal{I}_2$ such that $\mathcal{I}_0 \subseteq \mathcal{I}_2$, it holds that $\mathcal{I}_2 \models \phi$.

The following definition gives a name to the mapping that associates each logical FM to its corresponding extensional representation.

Definition 4.3 (The `ext` mapping). Let $(\mathcal{F}, \phi)$ be a FM in $\mathfrak{P}(X)$. We denote by $\mathtt{ext}((\mathcal{F}, \phi))$ (or $\mathtt{ext}(\mathcal{F}, \phi)$, for short) the extensional FM $(\mathcal{F}, \mathcal{P}) \in \mathfrak{C}(X)$ such that $\mathcal{P} = \{p \mid p \subseteq \mathcal{F} \text{ and } \mathcal{I}_p^{\mathcal{F}} \models \phi\}$. In particular, `ext` maps $\mathfrak{P}_{\mathrm{fin}}(X)$ to $\mathfrak{C}_{\mathrm{fin}}(X)$, and maps $\mathfrak{P}_{\mathrm{eql}}(X)$ to $\mathfrak{C}_{\mathrm{eql}}(X)$.

We denote by $\equiv$ the equivalence relation over FMs defined by: $(\mathcal{F}_1, \phi_1) \equiv (\mathcal{F}_2, \phi_2)$ if and only if both $\mathcal{F}_1 = \mathcal{F}_2$ and $\phi_1 \equiv \phi_2$. We write $[\mathfrak{P}(X)]$, $[\mathfrak{P}_{\mathrm{fin}}(X)]$ and $[\mathfrak{P}_{\mathrm{eql}}(X)]$ as short for the quotient sets $\mathfrak{P}(X)/\equiv$, $\mathfrak{P}_{\mathrm{fin}}(X)/\equiv$ and $\mathfrak{P}_{\mathrm{eql}}(X)/\equiv$, respectively.

Note that, if X has infinitely many elements and $(\mathcal{F}, \phi) \in \mathfrak{P}(X)$, then $\mathcal{F}$ may contain infinitely many features, while the propositional formula ϕ is syntactically finite (cf. Definition 2.1). Moreover, $\mathfrak{P}_{\mathrm{fin}}(X)$ has infinitely many elements (even when X is finite). It is also worth observing that, if X is finite, then $\mathfrak{P}(X)$ and $\mathfrak{P}_{\mathrm{fin}}(X)$ coincide and the quotient set $[\mathfrak{P}_{\mathrm{fin}}(X)]$ is finite. Moreover, for all $\Phi_1, \Phi_2 \in \mathfrak{P}(X)$, we have that: $\mathtt{ext}(\Phi_1) = \mathtt{ext}(\Phi_2)$ if and only if $\Phi_1 \equiv \Phi_2$.

All the finite FMs have a logical representation.

Theorem 4.4 (Completeness of logical representation for finite FMs). *For each $(\mathcal{F}, \mathcal{P}) \in \mathfrak{C}_{\mathrm{fin}}(X)$ there exists $(\mathcal{F}, \phi) \in \mathfrak{P}_{\mathrm{fin}}(X)$ such that $\mathtt{ext}(\mathcal{F}, \phi) = (\mathcal{F}, \mathcal{P})$.*

Proof. Take, for instance, disjunctive normal form $\phi = \bigvee_{p \in \mathcal{P}} \big((\wedge_{f \in p} f) \wedge (\wedge_{f \in \mathcal{F} \setminus p} \neg f)\big)$.

Given $[\Phi] \in [\mathfrak{P}(X)]$, we define (with an abuse of notation) $\mathtt{ext}([\Phi]) = \mathtt{ext}(\Phi)$. Then, we have that `ext` is an injection from $[\mathfrak{P}(X)]$ to $\mathfrak{C}(X)$, an injection from $[\mathfrak{P}_{\mathrm{eql}}(X)]$ to $\mathfrak{C}_{\mathrm{eql}}(X)$, and a bijection from $[\mathfrak{P}_{\mathrm{fin}}(X)]$ to $\mathfrak{C}_{\mathrm{fin}}(X)$.

As shown by the following example, if X has infinitely many elements, then there are FMs in $\mathfrak{C}(X) \setminus \mathfrak{C}_{\mathrm{fin}}(X)$ that have no logical representation.

Example 4.5 (FMs without a logical representation). Consider the set of features $\mathcal{F}_{\mathbb{N}} = \{f_1 \mid i \in \mathbb{N}\}$. Then the extensional FMs $(\mathcal{F}_{\mathbb{N}}, \{\{f_3\}\})$, $(\mathcal{F}_{\mathbb{N}}, \{\{f_n \mid n \text{ is even}\}\})$ (which has a single product with infinitely many features) and $(\mathcal{F}_{\mathbb{N}}, \{\{f_n\} \mid n \text{ is even}\})$ (which has infinitely many products with one feature each) have no logical representation.

Remark 4.6 (On $\mathfrak{P}_{\text{fin}}(X)$ and $\mathfrak{P}(X)$). It is worth observing that, since $\texttt{ext}(\mathcal{F}, \phi) = \texttt{ext}(\text{ftrs}(\phi), \phi) \bullet (\mathcal{F} \setminus \text{ftrs}(\phi), 2^{\mathcal{F} \setminus \text{ftrs}(\phi)})$ and the set $\text{ftrs}(\phi)$ is finite, then any infinite logical feature model (i.e., in $\mathfrak{P}(X) \setminus \mathfrak{P}_{\text{fin}}(X)$ with X infinite) is decomposable into a finite one (i.e., in $\mathfrak{P}_{\text{fin}}(X)$) and a "free" one (i.e., one where all the features are optional). Therefore, if X has infinitely many elements, then there are *infinitely* many elements of $\mathfrak{C}(X) \setminus \mathfrak{C}_{\text{fin}}(X)$ that do not have a logical representation.

4.2. Logical Characterization of the Lattices of FMs

The following theorem states that the FM fragment relation $\leq$ corresponds to (the converse of) logical consequence.

Theorem 4.7 (Logical characterization of $\leq$ on FMs). *Given $\Phi_1 = (\mathcal{F}_1, \phi_1)$ and $\Phi_2 = (\mathcal{F}_2, \phi_2)$ in $\mathfrak{P}(X)$, we write $\Phi_1 \leq \Phi_2$ to mean that both $\mathcal{F}_1 \subseteq \mathcal{F}_2$ and $\phi_2 \models \phi_1$ hold. Then: $\texttt{ext}(\Phi_1) \leq \texttt{ext}(\Phi_2)$ holds if and only if $\Phi_1 \leq \Phi_2$ holds.*

Proof. Let $\texttt{ext}(\mathcal{F}_i, \phi_i) = (\mathcal{F}_i, \mathcal{P}_i)$ s.t. $\mathcal{P}_i = \{p_i \mid p_i \subseteq \mathcal{F}_i \text{ and } \mathcal{I}_{p_i}^{\mathcal{F}_i} \models \phi_i\}$ for $i = 1, 2$. We have:

$\texttt{ext}(\Phi_1) \leq \texttt{ext}(\Phi_2)$
iff $\mathcal{F}_1 \subseteq \mathcal{F}_2$ and $\mathcal{P}_1 \supseteq \mathcal{P}_2|_{\mathcal{F}_1}$ (by Lemma 3.3)
iff $\mathcal{F}_1 \subseteq \mathcal{F}_2$ and $\{p_1 \subseteq \mathcal{F}_1 \mid \mathcal{I}_{p_1}^{\mathcal{F}_1} \models \phi_1\} \supseteq \{p_2 \cap \mathcal{F}_1 \mid p_i \subseteq \mathcal{F}_i \text{ and } \mathcal{I}_{p_2}^{\mathcal{F}_2} \models \phi_2\}$
iff $\mathcal{F}_1 \subseteq \mathcal{F}_2$ and, for all $p \in \mathcal{P}_2$, $\mathcal{I}_p^{\mathcal{F}_2} \models \phi_2$ implies $\mathcal{I}_p^{\mathcal{F}_2} \models \phi_1$
iff $\mathcal{F}_1 \subseteq \mathcal{F}_2$ and $\phi_2 \models \phi_1$
iff $\Phi_1 \leq \Phi_2$. □

The following theorem shows that the FM composition operator $\bullet$ corresponds to logic conjunction (cf. Sect. 2.3).

Theorem 4.8 (Logical characterization of $\bullet$ on FMs). *Given $\Phi_1 = (\mathcal{F}_1, \phi_1)$ and $\Phi_2 = (\mathcal{F}_2, \phi_2)$ in $\mathfrak{P}(X)$.*
We define: $\Phi_1 \bullet \Phi_2 = (\mathcal{F}_1 \cup \mathcal{F}_2, \phi_1 \wedge \phi_2)$. So, $\texttt{ext}(\Phi_1) \bullet \texttt{ext}(\Phi_2) = \texttt{ext}(\Phi_1 \bullet \Phi_2)$.

Proof. Let $\texttt{ext}(\mathcal{F}_i, \phi_i) = (\mathcal{F}_i, \mathcal{P}_i)$ s.t. $\mathcal{P}_i = \{p_i \mid p_i \subseteq \mathcal{F}_i \text{ and } \mathcal{I}_{p_i}^{\mathcal{F}_i} \models \phi_i\}$ for $i = 1, 2$.

$\texttt{ext}(\Phi_1) \bullet \texttt{ext}(\Phi_2) = (\mathcal{F}_3, \mathcal{P}_3)$
iff $\mathcal{F}_3 = \mathcal{F}_1 \cup \mathcal{F}_2$ and $\mathcal{P}_3 = \{p_1 \cup p_2 \mid p_1 \in \mathcal{P}_1, p_2 \in \mathcal{P}_2, p_1 \cap \mathcal{F}_2 = p_2 \cap \mathcal{F}_1\}$
iff $\mathcal{F}_3 = \mathcal{F}_1 \cup \mathcal{F}_2$ and $\mathcal{P}_3 = \{p_1 \cup p_2 \mid \mathcal{I}_{p_1}^{\mathcal{F}_1} \models \phi_1, \mathcal{I}_{p_2}^{\mathcal{F}_2} \models \phi_1, p_1 \cap \mathcal{F}_2 = p_2 \cap \mathcal{F}_1\}$
iff $\mathcal{F}_3 = \mathcal{F}_1 \cup \mathcal{F}_2$ and $\mathcal{P}_3 = \{p \mid p_1 \cup p_2 \subseteq p \text{ and } \mathcal{I}_p^{\mathcal{F}_3} \models \phi_1, \mathcal{I}_p^{\mathcal{F}_3} \models \phi_2\}$
iff $\mathcal{F}_3 = \mathcal{F}_1 \cup \mathcal{F}_2$ and $\mathcal{P}_3 = \{p \mid \mathcal{I}_p^{\mathcal{F}_3} \models \phi_1 \wedge \phi_2\}$
iff $(\mathcal{F}_3, \mathcal{P}_3) = \texttt{ext}(\Phi_1 \bullet \Phi_2)$. □

In order to provide a logical characterization of the meet operator $\star$ (c.f. Definition 3.5), we introduce an auxiliary notation expressing a logical encoding of the existentially quantified formula $\exists x_1. \cdots \exists x_n. \phi$, where ϕ is a propositional formula. Given $Y = \{x_1, ..., x_n\}$, we define:

$$\bigvee_{Y}(\phi) = \begin{cases} \phi & \text{if } Y = \emptyset, \\ \bigvee_{Y - \{x\}}\big((\phi[x := \mathsf{true}]) \vee (\phi[x := \mathsf{false}])\big) & \text{otherwise.} \end{cases}$$

Theorem 4.9 (Logical characterization $\star$ on FMs). *Given* $\Phi_1 = (\mathcal{F}_1, \phi_1)$ *and* $\Phi_2 = (\mathcal{F}_2, \phi_2)$ *in* $\mathfrak{P}(X)$*, we define:*

$$\Phi_1 \star \Phi_2 = \big(\mathcal{F}_1 \cap \mathcal{F}_2, \bigvee_{\mathrm{ftrs}(\phi_1)\setminus\mathcal{F}_2} (\phi_1) \vee \bigvee_{\mathrm{ftrs}(\phi_2)\setminus\mathcal{F}_1} (\phi_2)\big).$$

Then: $\mathtt{ext}(\Phi_1) \star \mathtt{ext}(\Phi_2) = \mathtt{ext}(\Phi_1 \star \Phi_2)$.

Proof. Let $\mathtt{ext}(\mathcal{F}_i, \phi_i) = (\mathcal{F}_i, \mathcal{P}_i)$ for $i = 1, 2$.
Since $\mathtt{ext}(\mathcal{F}_1, \phi_1) \star \mathtt{ext}(\mathcal{F}_2, \phi_2) = (\mathcal{F}_1 \cap \mathcal{F}_2, \mathcal{P}_1|_{\mathcal{F}_2} \cup \mathcal{P}_2|_{\mathcal{F}_1})$, we have that:

$\mathtt{ext}(\Phi_1) \star \mathtt{ext}(\Phi_2) = (\mathcal{F}_3, \mathcal{P}_3)$
iff $\mathcal{F}_3 = \mathcal{F}_1 \cap \mathcal{F}_2$ and $\mathcal{P}_3 = \{p_1 \cap \mathcal{F}_2 \mid \mathcal{I}_{p_1}^{\mathcal{F}_1} \models \phi_1\} \cup \{p_2 \cap \mathcal{F}_1 \mid \mathcal{I}_{p_2}^{\mathcal{F}_2} \models \phi_2\}$
iff $\mathcal{F}_3 = \mathcal{F}_1 \cap \mathcal{F}_2$ and $\mathcal{P}_3 = \{p_1 \cap \mathcal{F}_3 \mid \mathcal{I}_{p_1}^{\mathcal{F}_1} \models \phi_1\} \cup \{p_2 \cap \mathcal{F}_3 \mid \mathcal{I}_{p_2}^{\mathcal{F}_2} \models \phi_2\}$
iff $\mathcal{F}_3 = \mathcal{F}_1 \cap \mathcal{F}_2$ and $\mathcal{P}_3 = \mathcal{P}_1|_{\mathcal{F}_3} \cup \mathcal{P}_2|_{\mathcal{F}_3}$
iff $\mathcal{F}_3 = \mathcal{F}_1 \cap \mathcal{F}_2$ and, $p \in \mathcal{P}_3$ implies
either $\exists p_1$ s.t. $p = p_1 \cap \mathcal{F}_3$ and $\mathcal{I}_{p_1}^{\mathcal{F}_1} \models \phi_1$ or $\exists p_2$ s.t. $p = p_2 \cap \mathcal{F}_3$ and $\mathcal{I}_{p_2}^{\mathcal{F}_2} \models \phi_2$
iff $\mathcal{F}_3 = \mathcal{F}_1 \cap \mathcal{F}_2$ and, $p \in \mathcal{P}_3$ implies
either $\mathcal{I}_p^{\mathcal{F}_3} \models \bigvee_{\mathrm{ftrs}(\phi_1)\setminus\mathcal{F}_2}(\phi_1)$ or $\mathcal{I}_p^{\mathcal{F}_3} \models \bigvee_{\mathrm{ftrs}(\phi_2)\setminus\mathcal{F}_1}(\phi_2)$
iff $\mathcal{F}_3 = \mathcal{F}_1 \cap \mathcal{F}_2$ and, $p \in \mathcal{P}_3$ implies $\mathcal{I}_p^{\mathcal{F}_3} \models \bigvee_{\mathrm{ftrs}(\phi_1)\setminus\mathcal{F}_2}(\phi_1) \vee \bigvee_{\mathrm{ftrs}(\phi_2)\setminus\mathcal{F}_1}(\phi_2)$
iff $(\mathcal{F}_3, \mathcal{P}_3) = \mathtt{ext}(\Phi_1 \star \Phi_2)$. □

The following theorem states that the trivial FM $\mathbf{trivial}(\mathcal{F}) = (\mathcal{F}, 2^{\mathcal{F}})$ and the void FM $\mathbf{void}(\mathcal{F}) = (\mathcal{F}, \emptyset)$ correspond to true and false, respectively—recall that (see Theorem 3.8) $\mathbf{trivial}(\emptyset)$ is the bottom of the lattices $(\mathfrak{C}(X), \leq)$ and $(\mathfrak{C}_{\mathrm{fin}}(X), \leq)$, while $\mathbf{trivial}(X)$ is the bottom of the Boolean lattice $(\mathfrak{C}_{\mathrm{eql}}(X), \leq)$, and $\mathbf{void}(X)$ is the top of the lattice $(\mathfrak{C}(X), \leq)$ and of the Boolean lattice $(\mathfrak{C}_{\mathrm{eql}}(X), \leq)$ and, if X is finite, of the lattice $(\mathfrak{C}_{\mathrm{fin}}(X), \leq)$.

Theorem 4.10 (Logical characterization of trivial and void FMs). *Let* $(\mathcal{F}, \phi) \in \mathfrak{P}(X)$.

1. $\mathtt{ext}(\mathcal{F}, \phi) = \mathbf{trivial}(\mathcal{F}) = (\mathcal{F}, 2^{\mathcal{F}})$ *if and only if* $\phi \equiv \mathsf{true}$.
2. $\mathtt{ext}(\mathcal{F}, \phi) = \mathbf{void}(\mathcal{F}) = (\mathcal{F}, \emptyset)$ *if and only if* $\phi \equiv \mathsf{false}$.

Proof. *1.* Immediate, because true is satisfied by all interpretations. *2.* Immediate, because no interpretation satisfies false. □

The following theorem shows that the FM complement operator $(\cdot)^{\dagger}$ (introduced in Definition 3.7) corresponds to logical negation.

Theorem 4.11 (Logical characterization of $(\cdot)^{\dagger}$ on FMs). *Given* $\Phi = (\mathcal{F}, \phi)$ *in* $\mathfrak{P}(X)$*, we define:* $\overline{\Phi} = (\mathcal{F}, \neg\phi)$*. Then* $\overline{\mathtt{ext}(\Phi)} = \mathtt{ext}(\overline{\Phi})$.

Proof. Straightforward. □

Lemma 4.12 below provides a representation of logical disjunction in terms of a novel FM operator, that we denote by $+$. Then, Lemma 4.13 sheds some light on the Boolean lattice $\mathfrak{C}_{\mathrm{eql}}(X)$, by showing that on $\mathfrak{C}_{\mathrm{eql}}(X)$ the meet operator $\star$ and the operator $+$ coincide.

Lemma 4.12 (The operator + and its logical characterization). *Let Y, Z be two sets, we define: $Y \Cup Z = \{y \cup z \mid y \in Y, z \in Z\}$. Given two FMs $\mathbb{F}_1 = (\mathcal{F}_1, \mathcal{P}_1)$ and $\mathbb{F}_2 = (\mathcal{F}_2, \mathcal{P}_2)$ in $\mathfrak{C}(X)$, we define: $\mathbb{F}_1 + \mathbb{F}_2 = (\mathcal{F}_1 \cup \mathcal{F}_2, (\mathcal{P}_1 \Cup 2^{(\mathcal{F}_2 \setminus \mathcal{F}_1)}) \cup (\mathcal{P}_2 \Cup 2^{(\mathcal{F}_1 \setminus \mathcal{F}_2)}))$. Given $\Phi_1 = (\mathcal{F}_1, \phi_1)$ and $\Phi_2 = (\mathcal{F}_2, \phi_2)$ in $\mathfrak{P}(X)$, we define: $\Phi_1 + \Phi_2 = (\mathcal{F}_1 \cup \mathcal{F}_2, \phi_1 \vee \phi_2)$. Then: $\mathtt{ext}(\Phi_1) + \mathtt{ext}(\Phi_2) = \mathtt{ext}(\Phi_1 + \Phi_2)$.*

Proof. Let $\mathtt{ext}(\mathcal{F}_i, \phi_i) = (\mathcal{F}_i, \mathcal{P}_i)$ for $i = 1, 2$. We have that
$\mathtt{ext}(\Phi_1) + \mathtt{ext}(\Phi_2) = (\mathcal{F}_3, \mathcal{P}_3)$ where $\mathcal{P}_3 = \mathcal{F}_1 \cup \mathcal{F}_2$
iff $\mathcal{P}_3 = \{p_1 \Cup 2^{(\mathcal{F}_2 \setminus \mathcal{F}_1)}) \mid \mathcal{I}^{\mathcal{F}_1}_{p_1} \models \phi_1\} \cup \{p_2 \Cup 2^{(\mathcal{F}_1 \setminus \mathcal{F}_2)} \mid \mathcal{I}^{\mathcal{F}_2}_{p_2} \models \phi_2\}$
iff $\mathcal{P}_3 = \mathcal{F}_1 \cup \mathcal{F}_2$ and, $p \in \mathcal{P}_3$ implies
either $p \in \{p_1 \Cup 2^{(\mathcal{F}_2 \setminus \mathcal{F}_1)}) \mid \mathcal{I}^{\mathcal{F}_1}_{p_1} \models \phi_1\}$ or $p \in \{p_2 \Cup 2^{(\mathcal{F}_1 \setminus \mathcal{F}_2)} \mid \mathcal{I}^{\mathcal{F}_2}_{p_2} \models \phi_2\}$
iff $\mathcal{P}_3 = \mathcal{F}_1 \cup \mathcal{F}_2$ and, $p \in \mathcal{P}_3$ implies either $\mathcal{I}^{\mathcal{F}_3}_{p} \models \phi_1$ or $\mathcal{I}^{\mathcal{F}_3}_{p} \models \phi_2$
iff $\mathcal{P}_3 = \mathcal{F}_1 \cup \mathcal{F}_2$ and, $p \in \mathcal{P}_3$ implies $\mathcal{I}^{\mathcal{F}_3}_{p \cap \mathcal{F}_1} \models \phi_1 \vee \phi_2$
iff $(\mathcal{F}_3, \mathcal{P}_3) = \mathtt{ext}(\Phi_1 + \Phi_2)$. □

Lemma 4.13 (The operators $\star$ and + on $\mathfrak{C}_{\mathrm{eql}}(X)$). *Given two FMs $\mathbb{F}_1 = (X, \mathcal{P}_1)$ and $\mathbb{F}_2 = (X, \mathcal{P}_2)$ in $\mathfrak{C}_{\mathrm{eql}}(X)$, we have that: $\mathbb{F}_1 \star \mathbb{F}_2 = \mathbb{F}_1 + \mathbb{F}_2 = (X, \mathcal{P}_1 \cup \mathcal{P}_2)$.*

Proof. Straightforward from the definitions of $\star$ and +. □

Given $[\Phi_1], [\Phi_2] \in [\mathfrak{P}(X)]$, we define (with an abuse of notation): $[\Phi_1] \leq [\Phi_2]$ as $\Phi_1 \leq \Phi_2$, $[\Phi_1] \bullet [\Phi_2] = [\Phi_1 \bullet \Phi_2]$, $[\Phi_1] \star [\Phi_2] = [\Phi_1 \star \Phi_2]$, $[\Phi_1] + [\Phi_2] = [\Phi_1 + \Phi_2]$, and $\overline{[\Phi_1]} = [\overline{\Phi_1}]$. Recall that a *homomorphism* is a structure-preserving map between two algebraic structures of the same type (e.g., between two lattices), a *monomorphism* is an injective homomorphism, and an *isomorphism* is a bijective homomorphism.

Theorem 4.14 (`ext` is a lattice monomorphism). *Given a set X of features:*

1. *$([\mathfrak{P}(X)], \leq)$ is a bounded lattice with join $\bullet$, meet $\star$, bottom $[(\emptyset, \mathsf{true})]$ and top $[(X, \mathsf{false})]$. Moreover, $\mathtt{ext}$ is a bounded lattice monomorphism from $([\mathfrak{P}(X)], \leq)$ to $(\mathfrak{C}(X), \leq)$.*

2. *If X has infinitely many elements, then $[\mathfrak{P}_{\mathrm{fin}}(X)]$ is a sublattice of $[\mathfrak{P}(X)]$ with the same bottom and no top. Moreover, $\mathtt{ext}$ is a lattice isomorphism from $[\mathfrak{P}_{\mathrm{fin}}(X)]$ to $\mathfrak{C}_{\mathrm{fin}}(X)$.*

3. *$[\mathfrak{P}_{\mathrm{eql}}(X)]$ is a sublattice of $[\mathfrak{P}(X)]$ and it is a Boolean lattice with bottom $[(X, \mathsf{true})]$, same top of $[\mathfrak{P}(X)]$, complement $(\cdot)^{\dagger}$, and where the meet behaves like +. Moreover, $\mathtt{ext}$ is a Boolean lattice monomorphism from $[\mathfrak{P}_{\mathrm{eql}}(X)]$ to $\mathfrak{C}_{\mathrm{eql}}(X)$ and it is an isomorphism whenever X is finite.*

Proof. Straightforward from Theorems 3.8, 4.7-4.11 and Lemmas 3.4, 4.12 and 4.13. □

4.3. Logical Characterization of Slices and Interfaces

The following theorem provides a logical characterization of the slice operator.

Theorem 4.15 (Logical characterization of the operator Π_Y). *Let $\Phi = (\mathcal{F}, \phi)$ be in $\mathfrak{P}(X)$. We define: $\Pi_Y(\Phi) = (Y \cap \mathcal{F}, \curlyvee_{\mathrm{ftrs}(\phi)\setminus Y}(\phi))$. Then: $\Pi_Y(\mathit{ext}(\Phi)) = \mathit{ext}(\Pi_Y(\Phi))$.*

Proof. We have:

$$\begin{aligned}
&\Pi_Y(\mathtt{ext}(\Phi)) = (\mathcal{F}_0, \mathcal{P}_0)\\
&\text{iff } \mathcal{F}_0 = \mathcal{F} \cap Y \text{ and } \mathcal{P}_0 = \{p \mid \mathcal{I}_p^{\mathcal{F}} \models \phi\}|_Y\\
&\text{iff } \mathcal{F}_0 = \mathcal{F} \cap Y \text{ and } \mathcal{P}_0 = \{p \cap Y \mid \mathcal{I}_p^{\mathcal{F}} \models \phi\}\\
&\text{iff } \mathcal{F}_0 = \mathcal{F} \cap Y \text{ and } \mathcal{P}_0 = \{p \cap Y \mid \mathcal{I}_p^{\mathcal{F}\cap Y} \models \phi\}\\
&\text{iff } \mathcal{F}_0 = \mathcal{F} \cap Y \text{ and, } p_0 \in \mathcal{P}_0 \text{ implies } \mathcal{I}_{p_0}^{\mathcal{F}\cap Y} \models \curlyvee_{\mathrm{ftrs}(\phi)\setminus Y}(\phi)\\
&\text{iff } (\mathcal{F}_0, \mathcal{P}_0) = \mathtt{ext}(\Pi_Y(\Phi)).
\end{aligned}$$

□

The following corollary provides a logical characterization of the interface relation $\mathbb{F}_1 \preceq \mathbb{F}_2$ which is the same as the interpretation of the slice operator $\mathbb{F}_1 = \Pi_Y(\mathbb{F}_2)$ when Y are the features of $\mathbb{F}_1$ (cf. Theorem 4.15 and Remark 3.19).

Corollary 4.16 (Logical characterization of the relation $\preceq$). *Given $\Phi_1 = (\mathcal{F}_1, \phi_1)$ and $\Phi_2 = (\mathcal{F}_2, \phi_2)$ in $\mathfrak{P}(X)$, we write $\Phi_1 \preceq \Phi_2$ to mean that both $\mathcal{F}_1 \subseteq \mathcal{F}_2$ and $\phi_1 \equiv \curlyvee_{\mathrm{ftrs}(\phi_2)\setminus\mathcal{F}_1}(\phi_2)$ hold. Then: $\mathit{ext}(\Phi_1) \preceq \mathit{ext}(\Phi_2)$ holds if and only $\Phi_1 \preceq \Phi_2$ holds.*

Proof. We have:

$$\begin{aligned}
&(\mathcal{F}_1, \mathcal{P}_1) = \mathtt{ext}(\Phi_1) \preceq \mathtt{ext}(\Phi_2) = (\mathcal{F}_2, \mathcal{P}_2)\\
&\text{iff } \mathcal{F}_1 \subseteq \mathcal{F}_2 \text{ and } \mathcal{P}_1 = \mathcal{P}_2|_{\mathcal{F}_1} \qquad \text{(by Definition 3.18)}\\
&\text{iff } \mathcal{F}_1 \subseteq \mathcal{F}_2 \text{ and } \{p_1 \mid \mathcal{I}_{p_1}^{\mathcal{F}_1} \models \phi_1\} = \{p_2 \cap \mathcal{F}_1 \mid \mathcal{I}_{p_2}^{\mathcal{F}_2} \models \phi_2\}\\
&\text{iff } \mathcal{F}_1 \subseteq \mathcal{F}_2 \text{ and, for all } p \in \mathcal{P}_2, \text{ both } \phi_2 \models \phi_1 \text{ and } \phi_1 \models \curlyvee_{\mathrm{ftrs}(\phi_2)\setminus\mathcal{F}_1}(\phi_2)\\
&\text{iff } \mathcal{F}_1 \subseteq \mathcal{F}_2 \text{ and } \phi_1 \equiv \curlyvee_{\mathrm{ftrs}(\phi_2)\setminus\mathcal{F}_1}(\phi_2)\\
&\text{iff } \Phi_1 \preceq \Phi_2.
\end{aligned}$$

□

5. Logical Characterization of AFM Operations and Relations

In this section we follow a similar pattern to that of Section 4. In Section 5.1 we introduce the ingredients useful to relate logical and extensional AFM. In Section 5.2 we provide logical characterizations of extensional operations for AFMs. In Section 5.3, we provide logical characterization of AFM slices and interfaces.

5.1. Relating Extensional and Logical AFMs

The following definition introduces a notion for three different sets of logical AFMs (see Definition 2.9) over sets of features, attributes and domains (cf. Definition 3.1).

Definition 5.1 (Sets of logical AFMs). Let X be a set of features, let Y a set of attributes, let $\mathcal{D}$ be a set of domains, let $\ddot{\alpha}$ be a function of $Y \to X$, let $\ddot{\delta}$ be a function of $Y \to \mathcal{D}$ and let $\mathtt{d}$ is a function of $\mathcal{D} \to \cup\mathcal{D}$ such that $\mathtt{d} \propto \ddot{\delta}$. We denote:

- $\mathfrak{L}(X, Y, \ddot{\alpha}, \mathcal{D}, \ddot{\delta}, \mathtt{d})$ the set of the logical AFMs $(\mathcal{F}, \mathcal{A}, \alpha, \mathcal{D}, \delta, \mathtt{d}, \psi)$ such that $\mathcal{F} \subseteq X$, $\mathcal{A} \subseteq \ddot{\alpha}^{-1}(\mathcal{F})$, $\alpha = \ddot{\alpha}|_{\mathcal{A}}$ and $\delta = \ddot{\delta}|_{\mathcal{A}}$ (clearly, ψ is a constraint over $\mathcal{F}$ and $\mathcal{A}$);
- $\mathfrak{L}_{\text{fin}}(X, Y, \ddot{\alpha}, \mathcal{D}, \ddot{\delta}, \mathtt{d})$ the subset of the finite elements of $\mathfrak{L}(X, Y, \ddot{\alpha}, \mathcal{D}, \ddot{\delta}, \mathtt{d})$, namely the set of AFMs $(\mathcal{F}, \mathcal{A}, \alpha, \mathcal{D}, \delta, \psi) \in \mathfrak{L}(X, Y, \ddot{\alpha}, \mathcal{D}, \ddot{\delta})$ such that $\mathcal{F} \subseteq_{\text{fin}} X$ and $\mathcal{A} \subseteq_{\text{fin}} \ddot{\alpha}^{-1}(\mathcal{F})$; and
- $\mathfrak{L}_{\text{eql}}(X, Y, \ddot{\alpha}, \mathcal{D}, \ddot{\delta}, \mathtt{d})$ is the subset of $\mathfrak{L}(X, Y, \ddot{\alpha}, \mathcal{D}, \ddot{\delta}, \mathtt{d})$ including the AFMs $(\mathcal{F}, \mathcal{A}, \alpha, \mathcal{D}, \delta, \mathtt{d}, \psi)$ such that $\mathcal{F} = X$, $\mathcal{A} = Y$, $\alpha = \ddot{\alpha}$ and $\delta = \ddot{\delta}$.

We abuse the notation, by denoting $\text{ftrs}(\psi)$ the set of features occurring in the constraint ψ and we denote $\text{attr}(\psi)$ the set of attributes occurring in the constraint ψ. We recall that the interpretation of constraints in AFMs has been introduced in Definition 2.9.

Definition 5.2 (Interpretation for attributed-products). Let $\mathbb{A} = (\mathcal{F}, \mathcal{A}, \alpha, \mathcal{D}, \delta, \mathcal{V})$ be an extensional AFM and $(p, v) \in \mathcal{V}$. The interpretation $\mathcal{I}^{\mathcal{F},\mathcal{A},\alpha,\mathcal{D},\delta,\mathtt{d}}_{(p,v)}$ that *represents* the product (p, v) w.r.t the default-function $\mathtt{d}$ (i.e. a function of $\mathcal{D} \to \cup\mathcal{D}$ such that $\mathtt{d} \propto \delta$) is the function such that: $\mathcal{I}^{\mathcal{F},\mathcal{A},\alpha,\mathcal{D},\delta,\mathtt{d}}_{(p,v)}(x) = \mathit{true}$ if $x \in p$; $\mathcal{I}^{\mathcal{F},\mathcal{A},\alpha,\mathcal{D},\delta,\mathtt{d}}_{(p,v)}(x) = \mathit{false}$ if $x \in \mathcal{F} \setminus p$; and, if $a \in \text{dom}(v)$ then $\mathcal{I}^{\mathcal{F},\mathcal{A},\alpha,\mathcal{D},\delta,\mathtt{d}}_{(p,v)}(a) = v(a)$ otherwise $\mathcal{I}^{\mathcal{F},\mathcal{A},\alpha,\mathcal{D},\delta,\mathtt{d}}_{(p,v)}(a) = \mathtt{d}(\delta(a))$. For the sake of readability, we write $\mathcal{I}^{\mathbb{A},\mathtt{d}}_{(p,v)}$ to abbreviate $\mathcal{I}^{\mathcal{F},\mathcal{A},\alpha,\mathcal{D},\delta,\mathtt{d}}_{(p,v)}$.

It is worth to remark that it is natural to expect that the truth of constraints is invariant w.r.t the choice of default values, in accord to the discussion done in Section 2.2. We abuse the notation, by denoting $\text{dom}(\mathcal{I})$ the set of features and attributes subject of $\mathcal{I}$. Let $\mathbb{A} = (\mathcal{F}, \mathcal{A}, \alpha, \mathcal{D}, \delta, \mathcal{V})$ be an extensional AFM and $\mathtt{d}$ a default-function such that $\mathtt{d} \propto \delta$. We write that $\mathcal{I} \propto_{\mathbb{A}} \mathtt{d}$ whenever, if $a \in \text{dom}(\mathcal{I})$ and $\mathcal{I}(\alpha(a)) = \mathsf{false}$ then $\mathcal{I}(a) = \mathtt{d}(\delta(a))$). We write $\mathcal{I} \models^{\mathbb{A}}_{\mathtt{d}} \psi$ to mean that the formula ψ is true under the interpretation $\mathcal{I}$ (i.e., $\mathcal{I} \propto_{\mathbb{A}} \mathtt{d}$, $\text{ftrs}(\psi) \cup \text{attr}(\psi) \subseteq \text{dom}(\mathcal{I})$ and, the formula obtained from ψ evaluates true, after the following replacing: each feature x occurring in ψ by $\mathcal{I}(x)$, each attribute a by $\mathcal{I}(a)$). We write $\psi_1 \models^{\mathbb{A}}_{\mathtt{d}} \psi_2$ to mean that ψ_2 is is a logical consequence of ψ_1 (i.e., for all interpretations $\mathcal{I}$, if $\mathcal{I} \models^{\mathbb{A}}_{\mathtt{d}} \psi_1$ then $\mathcal{I} \models^{\mathbb{A}}_{\mathtt{d}} \psi_2$). We write $\psi_1 \equiv^{\mathbb{A}}_{\mathtt{d}} \psi_2$ to mean that ψ_1 and ψ_2 are logically equivalent (i.e., they are satisfied by the same interpretations with suitable domain).

The mapping associating a logical AFM to its corresponding extensional representation follows.

Definition 5.3 (The `ext` mapping for AFM). Let $\Psi = (\mathcal{F}, \mathcal{A}, \alpha, \mathcal{D}, \delta, \mathtt{d}, \psi)$ be logical AFM in $\mathfrak{L}(X, Y, \ddot{\alpha}, \mathcal{D}, \ddot{\delta}, \mathtt{d})$. For sake of readability, we write $\mathcal{I}^{\Psi}_{(p,v)}$ to

abbreviate $\mathcal{I}^{\mathcal{F},\mathcal{A},\alpha,\mathcal{D},\delta,\mathtt{d}}_{(p,v)}$ (cf. the similar abbreviation introduced at the end of Definition 5.2). We write $\mathbb{A} = \mathtt{ext}((\mathcal{F},\mathcal{A},\alpha,\mathcal{D},\delta,\mathtt{d},\psi))$ (or $\mathtt{ext}(\mathcal{F},\mathcal{A},\alpha,\mathcal{D},\delta,\mathtt{d},\psi)$ for short) to denote $(\mathcal{F},\mathcal{A},\alpha,\mathcal{D},\delta,\mathcal{V}) \in \mathfrak{A}(X,Y,\ddot{\alpha},\mathcal{D},\ddot{\delta})$ such that $\mathcal{V} = \{(p,v) \in 2^{\mathcal{F}} \times 2^{\alpha^{-1}(p)\to\cup\mathcal{D}} \mid v \propto \delta \text{ and } \mathcal{I}^{\Psi}_{(p,v)} \models_{\mathtt{d}} \psi\}$. It is straightforward that $\mathtt{ext}$ maps $\mathfrak{L}_{\text{fin}}(X,Y,\ddot{\alpha},\mathcal{D},\ddot{\delta},\mathtt{d})$, $\mathfrak{L}_{\text{eql}}(X,Y,\ddot{\alpha},\mathcal{D},\ddot{\delta},\mathtt{d})$ onto $\mathfrak{A}_{\text{fin}}(X,Y,\ddot{\alpha},\mathcal{D},\ddot{\delta})$, $\mathfrak{A}_{\text{eql}}(X,Y,\ddot{\alpha},\mathcal{D},\ddot{\delta})$ respectively.

Let $\Psi_i = (\mathcal{F}_i,\mathcal{A}_i,\alpha_i,\mathcal{D},\delta_i,\mathtt{d},\psi_i)$ be a logical AFM in $\mathfrak{L}(X,Y,\ddot{\alpha},\mathcal{D},\ddot{\delta},\mathtt{d})$ $i = 1,2$). We extend the logical equivalence on constraints to AFMs: we denote by $\equiv_{\mathtt{d}}$ the equivalence relation over AFMs defined by: $\Psi_1 \equiv_{\mathtt{d}} \Psi_2$ if and only if $\mathcal{F}_1 = \mathcal{F}_2$, $\mathcal{A}_1 = \mathcal{A}_2$, $\alpha_1 = \alpha_2$, $\delta_1 = \delta_2$ and $\psi_1 \equiv_{\mathtt{d}} \psi_2$. We write $[\mathfrak{L}(X,Y,\ddot{\alpha},\mathcal{D},\ddot{\delta},\mathtt{d})]$, $[\mathfrak{L}_{\text{fin}}(X,Y,\ddot{\alpha},\mathcal{D},\ddot{\delta},\mathtt{d})]$ and $[\mathfrak{L}_{\text{eql}}(X,Y,\ddot{\alpha},\mathcal{D},\ddot{\delta},\mathtt{d})]$ as short for $\mathfrak{L}(X,Y,\ddot{\alpha},\mathcal{D},\ddot{\delta},\mathtt{d})/\equiv_{\mathtt{d}}$, $\mathfrak{L}_{\text{fin}}(X,Y,\ddot{\alpha},\mathcal{D},\ddot{\delta},\mathtt{d})/\equiv_{\mathtt{d}}$ and $\mathfrak{L}_{\text{eql}}(X,Y,\ddot{\alpha},\mathcal{D},\ddot{\delta},\mathtt{d})/\equiv_{\mathtt{d}}$, respectively.

By construction, it is straightforward to see that each logical AFM identifies an extensional AFM. The following theorem shows that each finite extensional AFM has a logical representation.

Theorem 5.4 (Completeness of logical representation for finite AFMs). *For each $\mathbb{A} = (\mathcal{F},\mathcal{A},\alpha,\mathcal{D},\delta,\mathcal{V}) \in \mathfrak{A}_{\text{fin}}(X,Y,\ddot{\alpha},\mathcal{D},\ddot{\delta})$ there exists a logical AFM $\Psi = (\mathcal{F},\mathcal{A},\alpha,\mathcal{D},\delta,\mathtt{d},\psi)$ having the same attributed-products of $\mathbb{A}$.*

Proof. For each $p \in \mathcal{P}_{\mathcal{V}}$ we let ψ_p be the conjunctive clause $\big((\bigwedge_{f\in p} f) \wedge (\bigwedge_{f\in\mathcal{F}\setminus p} \neg f)\big)$. (If $p = \emptyset$ then $\psi_p = \mathtt{true}$.) For each $(p,v) \in \mathcal{V}$, let ψ^v_p be $\bigwedge_{a\in\text{dom}(v)} \big(a = v(a)\big)$. (If $\text{dom}(v) = \emptyset$ then ψ^v_p is $\mathtt{true}$). Let $\psi = \bigvee_{(p,v)\in\mathcal{V}} \big(\psi_p \wedge \psi^v_p\big)$. It is worth to note that the formula ψ is such that, for each (p,v) the values assigned to attributes in $\mathcal{A} \setminus \text{dom}(v)$ do not change its truth value. Therefore, it does not depend on $\mathtt{d}$ and the proof follows straightforwardly. □

5.2. Logical Characterization of the Lattices of AFMs

Theorem 5.5 (Logical characterization of $\leq$ on AFMs). *Let Ψ_i where $i = 1,2$ be the AFMs $(\mathcal{F}_i,\mathcal{A}_i,\alpha_i,\mathcal{D},\delta_i,\mathtt{d},\psi_i) \in \mathfrak{L}(X,Y,\ddot{\alpha},\mathcal{D},\ddot{\delta},\mathtt{d})$. We write $\Psi_1 \leq \Psi_2$ to mean that $\mathcal{F}_1 \subseteq \mathcal{F}_2$, $\mathcal{A}_1 \subseteq \mathcal{A}_2$, $\alpha_1 \subseteq \alpha_2$, $\delta_1 \subseteq \delta_2$ and $\psi_2 \models_{\mathtt{d}} \psi_1$. Then: $\mathtt{ext}(\Psi_1) \leq \mathtt{ext}(\Psi_2)$ holds if and only $\Psi_1 \leq \Psi_2$ holds.*

Proof. Let $\mathtt{ext}(\Psi_i) = (\mathcal{F}_i,\mathcal{A}_i,\alpha_i,\mathcal{D},\delta_i,\mathcal{V}_i)$ where $i = 1,2$ such that $\mathcal{V}_i = \{(p_i,v_i) \in 2^{\mathcal{F}_i} \times 2^{\alpha_i^{-1}(p_i)\to\cup\mathcal{D}} \mid v_i \propto \delta_i \text{ and } \mathcal{I}^{\Psi_i}_{(p_i,v_i)} \models_{\mathtt{d}} \psi_i\}$.

We have: $\texttt{ext}(\Psi_1) \leq \texttt{ext}(\Psi_2)$

iff $\left(\begin{array}{l} \mathcal{F}_1 \subseteq \mathcal{F}_2, \mathcal{A}_1 \subseteq \mathcal{A}_2, \alpha_1 \subseteq \alpha_2, \delta_1 \subseteq \delta_2, \\ \mathcal{V}_1 \supseteq \{(p_2|_{\mathcal{F}_1}, v_2|_{\mathcal{A}_1}) \mid (p_2, v_2) \in \mathcal{V}_2\} \end{array}\right)$ by Lemma 3.11,

iff $\left(\begin{array}{l} \mathcal{F}_1 \subseteq \mathcal{F}_2, \mathcal{A}_1 \subseteq \mathcal{A}_2, \alpha_1 \subseteq \alpha_2, \delta_1 \subseteq \delta_2 \\ \{(p_1, v_1) \mid \mathcal{I}^{\Psi_1}_{(p_1,v_1)} \models_{\mathtt{d}} \psi_1\} \supseteq \{(p_2|_{\mathcal{F}_1}, v_2|_{\mathcal{A}_1}) \mid \mathcal{I}^{\Psi_2}_{(p_2,v_2)} \models_{\mathtt{d}} \psi_2\} \end{array}\right)$

iff $\left(\begin{array}{l} \mathcal{F}_1 \subseteq \mathcal{F}_2, \mathcal{A}_1 \subseteq \mathcal{A}_2, \alpha_1 \subseteq \alpha_2, \delta_1 \subseteq \delta_2, \\ \mathcal{I}^{\Psi_2}_{(p_2,v_2)} \models_{\mathtt{d}} \psi_2 \text{ implies } \mathcal{I}^{\Psi_1}_{(p_2|_{\mathcal{F}_1}, v_2|_{\mathcal{A}_1})} \models_{\mathtt{d}} \psi_1 \end{array}\right)$

because $\left(\begin{array}{l} \text{ftrs}(\psi_1) \subseteq \mathcal{F}_1 \subseteq \mathcal{F}_2 \ , \ \text{attr}(\psi_1) \subseteq \mathcal{A}_1 \subseteq \mathcal{A}_2 \\ \text{and, in particular, } \alpha_1^{-1}(\mathcal{F}_1) \subseteq \alpha_2^{-1}(\mathcal{F}_2) \end{array}\right)$

iff $\mathcal{F}_1 \subseteq \mathcal{F}_2, \mathcal{A}_1 \subseteq \mathcal{A}_2, \alpha_1 \subseteq \alpha_2, \delta_1 \subseteq \delta_2,$ and $\psi_2 \models_{\mathtt{d}} \psi_1$

iff $\Psi_1 \leq \Psi_2$. □

Theorem 5.6 (Logical characterization of • on AFMs). *Let us assume Ψ_i where $i = 1, 2$ be the AFMs $(\mathcal{F}_i, \mathcal{A}_i, \alpha_i, \mathcal{D}, \delta_i, \mathtt{d}, \psi_i) \in \mathfrak{AL}(X, Y, \ddot{\alpha}, \mathcal{D}, \ddot{\delta}, \mathtt{d})$. We define $\Psi_1 \bullet \Psi_2 = (\mathcal{F}_1 \cup \mathcal{F}_2, \mathcal{A}_1 \cup \mathcal{A}_2, \alpha_1 \oplus \alpha_2, \mathcal{D}, \delta_1 \oplus \delta_2, \mathtt{d}, \psi_1 \wedge \psi_2)$. Then: $\texttt{ext}(\Psi_1) \bullet \texttt{ext}(\Psi_2) = \texttt{ext}(\Psi_1 \bullet \Psi_2)$.*

Proof. Let $\texttt{ext}(\Psi_i) = (\mathcal{F}_i, \mathcal{A}_i, \alpha_i, \mathcal{D}, \delta_i, \mathcal{V}_i)$ where $i = 1, 2$ such that $\mathcal{V}_i = \{(p_i, v_i) \in 2^{\mathcal{F}_i} \times 2^{\alpha_i^{-1}(p_i) \to \cup \mathcal{D}} \mid \text{dom}(v_i) = \alpha_i^{-1}(p_i) \ , \ v_i \propto \delta_i \ , \ \mathcal{I}^{\Psi_i}_{(p_i,v_i)} \models_{\mathtt{d}} \psi_i\}$.

$\texttt{ext}(\Psi_1) \bullet \texttt{ext}(\Psi_2) = (\mathcal{F}_3, \mathcal{A}_3, \alpha_3, \mathcal{D}, \delta_3, \mathcal{V}_3)$

iff $\left(\begin{array}{l} \mathcal{F}_3 = \mathcal{F}_1 \cup \mathcal{F}_2, \mathcal{A}_3 = \mathcal{A}_1 \cup \mathcal{A}_2, \alpha_3 = \alpha_1 \oplus \alpha_2, \delta_3 = \delta_1 \oplus \delta_2, \\ \mathcal{V}_3 = \{(p_1 \cup p_2, v_1 \oplus v_2) \mid (p_i, v_i) \in \mathcal{V}_i \ , \ p_1 \cap \mathcal{F}_2 = p_2 \cap \mathcal{F}_1 \ , \ v_1 \ncong v_2\} \end{array}\right)$

iff $\left(\begin{array}{l} \mathcal{F}_3 = \mathcal{F}_1 \cup \mathcal{F}_2, \mathcal{A}_3 = \mathcal{A}_1 \cup \mathcal{A}_2, \alpha_3 = \alpha_1 \oplus \alpha_2, \delta_3 = \delta_1 \oplus \delta_2, \\ \text{and } \mathcal{V}_3 = \left\{ (p_1 \cup p_2, v_1 \oplus v_2) \middle| \begin{array}{l} p_1 \cap \mathcal{F}_2 = p_2 \cap \mathcal{F}_1, \ v_1 \ncong v_2 \\ v_i \propto \delta_i \ , \ \mathcal{I}^{\Psi_i}_{(p_i,v_i)} \models_{\mathtt{d}} \psi_i \end{array} \right\} \end{array}\right)$

iff $\left(\begin{array}{l} \mathcal{F}_3 = \mathcal{F}_1 \cup \mathcal{F}_2, \mathcal{A}_3 = \mathcal{A}_1 \cup \mathcal{A}_2, \alpha_3 = \alpha_1 \oplus \alpha_2, \delta_3 = \delta_1 \oplus \delta_2, \\ \text{and } \mathcal{V}_3 = \{(p, v) \mid \alpha_3^{-1}(p) = \text{dom}(v), \ v \propto \delta_3, \ \mathcal{I}^{\Psi_i}_{(p|_{\mathcal{F}_i}, v|_{\mathcal{A}_i})} \models_{\mathtt{d}} \psi_i\} \end{array}\right)$

iff $\left(\begin{array}{l} \mathcal{F}_3 = \mathcal{F}_1 \cup \mathcal{F}_2, \mathcal{A}_3 = \mathcal{A}_1 \cup \mathcal{A}_2, \alpha_3 = \alpha_1 \oplus \alpha_2, \delta_3 = \delta_1 \oplus \delta_2, \text{and} \\ \mathcal{V}_3 = \{(p, v) \mid \alpha_3^{-1}(p) = \text{dom}(v) \ , \ v \propto \delta_3 \ , \ \mathcal{I}^{\mathcal{F}_3, \mathcal{A}_3, \alpha_3, \mathcal{D}, \delta_3, \mathtt{d}}_{(p,v)} \models_{\mathtt{d}} \psi_1 \wedge \psi_2\} \end{array}\right)$

iff $\left(\begin{array}{l} \mathcal{F}_3 = \mathcal{F}_1 \cup \mathcal{F}_2, \mathcal{A}_3 = \mathcal{A}_1 \cup \mathcal{A}_2, \alpha_3 = \alpha_1 \oplus \alpha_2, \delta_3 = \delta_1 \oplus \delta_2, \\ \mathcal{V}_3 = \left\{ (p, v) \in 2^{\mathcal{F}_3} \times 2^{\alpha_3^{-1}(p) \to \cup \mathcal{D}} \middle| \begin{array}{l} v \propto \delta_3 \text{ and} \\ \mathcal{I}^{\mathcal{F}_3, \mathcal{A}_3, \alpha_3, \mathcal{D}, \delta_3, \mathtt{d}}_{(p,v)} \models_{\mathtt{d}} \psi_1 \wedge \psi_2 \end{array} \right\} \end{array}\right)$

iff $\mathcal{F}_3 = \texttt{ext}(\Psi_1 \bullet \Psi_2)$. □

In order to provide a logical characterization of the meet operator $\star$ for AFMs (c.f. Definition 3.13), we need to adapt the encoding for the existentially quantified features introduced for traditional feature. Let $(\mathcal{F}, \mathcal{A}, \alpha, \mathcal{D}, \delta, \mathtt{d}, \psi)$ be a logical AFM in $\mathfrak{AL}(X, Y, \ddot{\alpha}, \mathcal{D}, \ddot{\delta}, \mathtt{d})$. If $Y = \{x_1, ..., x_n\}$ is a set of features then we define:

$$\bigvee_Y(\psi) = \begin{cases} \psi & \text{if } Y = \emptyset, \\ \bigvee_{Y - \{x\}}(\psi^{[x]}_{\mathsf{true}} \vee \psi^{[x]}_{\mathsf{false}}) & \text{otherwise;} \end{cases}$$

where $\{a_1, \ldots, a_n\} = \alpha^{-1}(x) \cap \text{attr}(\psi)$, $D_i = \delta(a_i)$, so

$$\psi^{[x]}_{\mathsf{true}} = (\overrightarrow{\exists a_i : D_i}.\psi)[x := \mathsf{true}] \quad \text{and} \quad \psi^{[x]}_{\mathsf{false}} = \psi[\overrightarrow{a_i := \mathtt{d}(D_i)}, x := \mathsf{false}] \ .$$

Theorem 5.7 (Logical characterization $\star$ on AFMs). *Let us assume Ψ_i be two AFMs $(\mathcal{F}_i, \mathcal{A}_i, \alpha_i, \mathcal{D}, \delta_i, \mathtt{d}, \psi_i) \in \mathfrak{L}(X, Y, \ddot{\alpha}, \mathcal{D}, \ddot{\delta}, \mathtt{d})$ where $i = 1, 2$. We define $\Psi_1 \star \Psi_2 = (\mathcal{F}_1 \cap \mathcal{F}_2, \mathcal{A}_1 \cap \mathcal{A}_2, \alpha_1 \ominus \alpha_2, \mathcal{D}, \delta_1 \ominus \delta_2, \curlyvee_{\mathcal{F}_1 \setminus \mathcal{F}_2}(\psi_1) \vee \curlyvee_{\mathcal{F}_2 \setminus \mathcal{F}_1}(\psi_2))$. Then: $\mathtt{ext}(\Psi_1) \star \mathtt{ext}(\Psi_2) = \mathtt{ext}(\Psi_1 \star \Psi_2)$.*

Proof. Let $\mathtt{ext}(\Psi_i) = (\mathcal{F}_i, \mathcal{A}_i, \alpha_i, \mathcal{D}, \delta_i, \mathcal{V}_i)$ where $i = 1, 2$ such that $\mathcal{V}_i = \{(p_i, v_i) \in 2^{\mathcal{F}_i} \times 2^{\alpha_i^{-1}(p) \to \cup \mathcal{D}} \mid v_i \propto \delta_i \text{ and } \mathcal{I}^{\Psi_i}_{(p_i, v_i)} \models_{\mathtt{d}} \psi_i\}$.

$$
\begin{aligned}
&\mathtt{ext}(\Psi_1) \star \mathtt{ext}(\Psi_2) = (\mathcal{F}_3, \mathcal{A}_3, \alpha_3, \mathcal{D}, \delta_3, \mathcal{V}_3) \\
&\text{iff} \left(\begin{array}{l} \mathcal{F}_3 = \mathcal{F}_1 \cap \mathcal{F}_2, \mathcal{A}_3 = \mathcal{A}_1 \cap \mathcal{A}_2, \alpha_3 = \alpha_1 \ominus \alpha_2, \delta_3 = \delta_1 \ominus \delta_2 \text{ and} \\ \mathcal{V}_3 = \{(p_1|_{\mathcal{F}_2}, v_1|_{\mathcal{A}_2}) \mid (p_1, v_1) \in \mathcal{V}_1\} \cup \{(p_2|_{\mathcal{F}_1}, v_2|_{\mathcal{A}_1}) \mid (p_2, v_2) \in \mathcal{V}_2\} \end{array} \right) \\
&\text{iff} \left(\begin{array}{l} \mathcal{F}_3 = \mathcal{F}_1 \cap \mathcal{F}_2, \mathcal{A}_3 = \mathcal{A}_1 \cap \mathcal{A}_2, \alpha_3 = \alpha_1 \ominus \alpha_2, \delta_3 = \delta_1 \ominus \delta_2 \text{ and} \\ \mathcal{V}_3 = \bigcup_{i=0,1} \left\{(p_i|_{\mathcal{F}_3}, v_i|_{\mathcal{A}_3}) \mid (p_i, v_i) \in \mathcal{V}_i\right\} \end{array} \right) \\
&\text{iff} \left(\begin{array}{l} \mathcal{F}_3 = \mathcal{F}_1 \cap \mathcal{F}_2, \mathcal{A}_3 = \mathcal{A}_1 \cap \mathcal{A}_2, \alpha_3 = \alpha_1 \ominus \alpha_2, \delta_3 = \delta_1 \ominus \delta_2 \text{ and} \\ \mathcal{V}_3 = \bigcup_{i=0,1} \left\{(p_i|_{\mathcal{F}_3}, v_i|_{\mathcal{A}_3}) \mid v_i \propto \delta_i \text{ and } \mathcal{I}^{\mathcal{F}_i, \mathcal{A}_i, \alpha_i, \mathcal{D}, \delta_i, \mathtt{d}}_{(p_i, v_i)} \models_{\mathtt{d}} \psi_i\right\} \end{array} \right) \\
&\text{iff} \left(\begin{array}{l} \mathcal{F}_3 = \mathcal{F}_1 \cap \mathcal{F}_2, \mathcal{A}_3 = \mathcal{A}_1 \cap \mathcal{A}_2, \alpha_3 = \alpha_1 \ominus \alpha_2, \delta_3 = \delta_1 \ominus \delta_2 \text{ and} \\ \mathcal{V}_3 = \bigcup_{i=0,1} \left\{(p_i|_{\mathcal{F}_3}, v_i|_{\mathcal{A}_3}) \mid v_i \propto \delta_i \text{ and } \mathcal{I}^{\mathcal{F}_i, \mathcal{A}_i, \alpha_i, \mathcal{D}, \delta_i, \mathtt{d}}_{(p_i, v_i)} \models_{\mathtt{d}} \psi_i\right\} \end{array} \right) \\
&\text{iff} \left(\begin{array}{l} \mathcal{F}_3 = \mathcal{F}_1 \cap \mathcal{F}_2, \mathcal{A}_3 = \mathcal{A}_1 \cap \mathcal{A}_2, \alpha_3 = \alpha_1 \ominus \alpha_2, \delta_3 = \delta_1 \ominus \delta_2 \text{ and} \\ (p_3, v_3) \in \mathcal{V}_3 \text{ implies either } \mathcal{I}^{\mathcal{F}_3, \mathcal{A}_3, \alpha_3, \mathcal{D}, \delta_3, \mathtt{d}}_{(p_3, v_3)} \models_{\mathtt{d}} \curlyvee_{\mathrm{ftrs}(\psi_1) \setminus \mathcal{F}_2}(\psi_1), \\ \qquad\qquad \text{or } \mathcal{I}^{\mathcal{F}_3, \mathcal{A}_3, \alpha_3, \mathcal{D}, \delta_3, \mathtt{d}}_{(p_3, v_3)} \models_{\mathtt{d}} \curlyvee_{\mathrm{ftrs}(\psi_2) \setminus \mathcal{F}_1}(\psi_2) \end{array} \right) \\
&\text{iff} \left(\begin{array}{l} \mathcal{F}_3 = \mathcal{F}_1 \cap \mathcal{F}_2, \mathcal{A}_3 = \mathcal{A}_1 \cap \mathcal{A}_2, \alpha_3 = \alpha_1 \ominus \alpha_2, \delta_3 = \delta_1 \ominus \delta_2 \text{ and} \\ (p_3, v_3) \in \mathcal{V}_3 \text{ implies} \\ \qquad \mathcal{I}^{\mathcal{F}_3, \mathcal{A}_3, \alpha_3, \mathcal{D}, \delta_3, \mathtt{d}}_{(p_3, v_3)} \models_{\mathtt{d}} \left(\curlyvee_{\mathrm{ftrs}(\phi_1) \setminus \mathcal{F}_2}(\psi_1) \vee \curlyvee_{\mathrm{ftrs}(\phi_1) \setminus \mathcal{F}_2}(\psi_2) \right) \end{array} \right) \\
&\text{iff } (\mathcal{F}_3, \mathcal{A}_3, \alpha_3, \mathcal{D}, \delta_3, \mathcal{V}_3) = \mathtt{ext}(\Psi_1 \star \Psi_2).
\end{aligned}
$$

□

The following theorem states that the trivial AFMs $\mathbf{trivial}(\mathcal{F}, \mathcal{A}, \alpha, \mathcal{D}, \delta) = (\mathcal{F}, \mathcal{A}, \alpha, \mathcal{D}, \delta, \mathbf{all}(\mathcal{F}, \mathcal{A}, \alpha, \mathcal{D}, \delta))$ and the void AFMs $\mathbf{void}(\mathcal{F}, \mathcal{A}, \alpha, \mathcal{D}, \delta) = (\mathcal{F}, \mathcal{A}, \alpha, \mathcal{D}, \delta, \emptyset)$ correspond to true and false, respectively—recall that (see Theorem 3.16) $\mathbf{trivial}(X, Y, \ddot{\alpha}, \mathcal{D}, \ddot{\delta}, \emptyset)$ is the bottom of the lattices $(\mathfrak{C}(X, Y, \ddot{\alpha}, \mathcal{D}, \ddot{\delta}), \leq)$ and $(\mathfrak{C}_{\mathrm{fin}}(X, Y, \ddot{\alpha}, \mathcal{D}, \ddot{\delta}), \leq)$, while $\mathbf{trivial}(X, Y, \ddot{\alpha}, \mathcal{D}, \ddot{\delta})$ is the bottom of the Boolean lattice $(\mathfrak{C}_{\mathrm{eql}}(X, Y, \ddot{\alpha}, \mathcal{D}, \ddot{\delta}), \leq$), and $\mathbf{void}(X, Y, \ddot{\alpha}, \mathcal{D}, \ddot{\delta})$ is the top of the lattice $(\mathfrak{C}(X, Y, \ddot{\alpha}, \mathcal{D}, \ddot{\delta}), \leq)$ and of the Boolean lattice $(\mathfrak{C}_{\mathrm{eql}}(X, Y, \ddot{\alpha}, \mathcal{D}, \ddot{\delta}), \leq)$ and, if X is finite, of the lattice $(\mathfrak{C}_{\mathrm{fin}}(X), \leq)$.

Theorem 5.8 (Logical characterization of trivial and void AFM). *Let us assume $(\mathcal{F}, \mathcal{A}, \alpha, \mathcal{D}, \delta, \mathtt{d}, \psi) \in \mathfrak{L}(X, Y, \ddot{\alpha}, \mathcal{D}, \ddot{\delta}, \mathtt{d})$.*

1. *$\mathtt{ext}(\mathcal{F}, \mathcal{A}, \alpha, \mathcal{D}, \delta, \mathtt{d}, \psi) = (\mathcal{F}, \mathcal{A}, \alpha, \mathcal{D}, \delta, \mathbf{all}(\mathcal{F}, \mathcal{A}, \alpha, \mathcal{D}, \delta))$ iff $\psi \equiv_{\mathtt{d}}$ true.*
2. *$\mathtt{ext}(\mathcal{F}, \mathcal{A}, \alpha, \mathcal{D}, \delta, \mathtt{d}, \psi) = (\mathcal{F}, \mathcal{A}, \alpha, \mathcal{D}, \delta, \emptyset)$ if and only if $\psi \equiv_{\mathtt{d}}$ false.*

Proof. *1.* Immediate, because true is satisfied by all interpretations. *2.* Immediate, because no interpretation satisfies false. □

The following theorem shows that the AFM complement operator $(\cdot)^{\dagger}$ (introduced in Definition 3.15) corresponds to logical negation.

Theorem 5.9 (Logical characterization of $(\cdot)^\dagger$ on AFMs).
Let $\Psi = (X, Y, \ddot{\alpha}, \mathcal{D}, \ddot{\delta}, \mathtt{d}, \psi) \in \mathfrak{L}(X, Y, \ddot{\alpha}, \mathcal{D}, \ddot{\delta}, \mathtt{d})$.
If $\Psi^\dagger = (X, Y, \ddot{\alpha}, \mathcal{D}, \ddot{\delta}, \neg\psi)$ *then* $\mathit{ext}(\Psi)^\dagger = \mathit{ext}(\Psi^\dagger)$.

Proof. Traditional semantics of classical logic is bivalent, meaning that all formulas are interpreted in true or false. More precisely, each interpretation makes true a formula and false its negation. Straightforwardly, this fact holds for our interpretations too, namely $\mathtt{ext}(\Psi) \cup \mathtt{ext}(\Psi)^\dagger = \mathbf{all}(\mathcal{F}, \mathcal{A}, \alpha, \mathcal{D}, \delta)$. We just note that $\mathbf{all}(\mathcal{F}, \mathcal{A}, \alpha, \mathcal{D}, \delta)$ does not include all possible classical interpretations, but only interpretations that satisfy some constraints on attributes relative to selected features. □

Lemma 5.10 characterizes the logical disjunction in terms of a the operator $+$. Then, Lemma 5.11 sheds some light on the Boolean lattice , by showing that on $\mathfrak{L}_{\mathrm{eql}}(X, Y, \ddot{\alpha}, \mathcal{D}, \ddot{\delta}, \mathtt{d})$ the meet operator $\star$ and the operator $+$ coincide.

Lemma 5.10 (The operator $+$ on AFMs and its logical characterization).
Let $\mathbb{A}_i$ *be two AFMs* $(\mathcal{F}_i, \mathcal{A}_i, \alpha_i, \mathcal{D}, \delta_i, \mathcal{V}_i) \in \mathfrak{A}(X, Y, \ddot{\alpha}, \mathcal{D}, \ddot{\delta})$ *where* $i = 1, 2$, *we define* $\mathbb{A}_1 + \mathbb{A}_2 = (\mathcal{F}_1 \cup \mathcal{F}_2, \mathcal{A}_1 \cup \mathcal{A}_2, \alpha_1 \oplus \alpha_2, \mathcal{D}, \delta_1 \oplus \delta_2, \mathcal{V}_1^{\boxplus} \cup \mathcal{V}_2^{\boxplus})$ *where* $i^\bullet = 3 - i$ *and* $\mathcal{V}_i^{\boxplus} = \{(p_i \cup q, v_i \cup u) \mid (p_i, v_i) \in \mathcal{V}_i$ *and* $(q, u) \in \mathbf{all}(\mathcal{F}_{i^\bullet}, \mathcal{A}_{i^\bullet}, \alpha_{i^\bullet}, \mathcal{D}, \delta_{i^\bullet})\}$.
Let Ψ_i *be logical AFMs* $(\mathcal{F}_i, \mathcal{A}_i, \alpha_i, \mathcal{D}, \delta_i, \mathtt{d}, \psi_i) \in \mathfrak{L}(X, Y, \ddot{\alpha}, \mathcal{D}, \ddot{\delta}, \mathtt{d})$ *where* $i = 1, 2$, *we define* $\Psi_1 + \Psi_2 = (\mathcal{F}_1 \cup \mathcal{F}_2, \mathcal{A}_1 \cup \mathcal{A}_2, \alpha_1 \oplus \alpha_2, \mathcal{D}, \delta_1 \oplus \delta_2, \psi_1 \vee \psi_2)$. *Then:* $\mathit{ext}(\Psi_1) + \mathit{ext}(\Psi_2) = \mathit{ext}(\Psi_1 + \Psi_2)$.

Proof. Let $\mathtt{ext}(\Psi_i) = (\mathcal{F}_i, \mathcal{A}_i, \alpha_i, \mathcal{D}, \delta_i, \mathcal{V}_i)$ where $i = 1, 2$ such that $\mathcal{V}_i = \{(p_i, v_i) \in 2^{\mathcal{F}_i} \times 2^{\alpha_i^{-1}(p_i) \to \cup \mathcal{D}} \mid v_i \propto \delta_i$ and $\mathcal{I}^{\Psi_i}_{(p_i, v_i)} \models_{\mathtt{d}} \psi_i\}$.

$$
\begin{aligned}
&\mathtt{ext}(\Psi_1) + \mathtt{ext}(\Psi_2) = (\mathcal{F}_3, \mathcal{A}_3, \alpha_3, \mathcal{D}, \delta_3, \mathcal{V}_3) \text{ where } \mathcal{V}_3 = \mathcal{V}_1^{\boxplus} \cup \mathcal{V}_2^{\boxplus} \\
&\text{iff} \left(\begin{array}{l} \mathcal{F}_3 = \mathcal{F}_1 \cup \mathcal{F}_2, \mathcal{A}_3 = \mathcal{A}_1 \cup \mathcal{A}_2, \alpha_3 = \alpha_1 \oplus \alpha_2, \delta_3 = \delta_1 \oplus \delta_2, \text{ and} \\ \mathcal{V}_i^{\boxplus} = \{(p_i \cup q, v_i \cup u) \mid (p_i, v_i) \in \mathcal{V}_i \text{ and } (q, u) \in \mathbf{all}(\mathcal{F}_{i^\bullet}, \mathcal{A}_{i^\bullet}, \alpha_{i^\bullet}, \mathcal{D}, \delta_{i^\bullet})\} \end{array} \right) \\
&\text{iff} \left(\begin{array}{l} \mathcal{F}_3 = \mathcal{F}_1 \cup \mathcal{F}_2, \mathcal{A}_3 = \mathcal{A}_1 \cup \mathcal{A}_2, \alpha_3 = \alpha_1 \oplus \alpha_2, \delta_3 = \delta_1 \oplus \delta_2, \text{ and} \\ \mathcal{V}_i^{\boxplus} = \left\{ (p_i \cup q, v_i \cup u) \;\middle|\; \begin{array}{l} v_i \propto \delta_i \;,\; \mathcal{I}^{\mathcal{F}_i, \mathcal{A}_i, \alpha_i, \mathcal{D}, \delta_i, \mathtt{d}}_{(p_i, v_i)} \models_{\mathtt{d}} \psi_i \text{ and} \\ (q, u) \in \mathbf{all}(\mathcal{F}_{i^\bullet}, \mathcal{A}_{i^\bullet}, \alpha_{i^\bullet}, \mathcal{D}, \delta_{i^\bullet}) \end{array} \right\} \end{array} \right) \\
&\text{iff} \left(\begin{array}{l} \mathcal{F}_3 = \mathcal{F}_1 \cup \mathcal{F}_2, \mathcal{A}_3 = \mathcal{A}_1 \cup \mathcal{A}_2, \alpha_3 = \alpha_1 \oplus \alpha_2, \delta_3 = \delta_1 \oplus \delta_2, \text{ and} \\ (p_3, v_3) \in \mathcal{V}_3 \text{ implies, either } \mathcal{I}^{\mathcal{F}_3, \mathcal{A}_3, \alpha_3, \mathcal{D}, \delta_3, \mathtt{d}}_{(p_3, v_3)} \models_{\mathtt{d}} \psi_1 \\ \qquad\qquad\qquad\qquad \text{or } \mathcal{I}^{\mathcal{F}_3, \mathcal{A}_3, \alpha_3, \mathcal{D}, \delta_3, \mathtt{d}}_{(p_3, v_3)} \models_{\mathtt{d}} \psi_2 \end{array} \right) \\
&\text{iff} \left(\begin{array}{l} \mathcal{F}_3 = \mathcal{F}_1 \cup \mathcal{F}_2, \mathcal{A}_3 = \mathcal{A}_1 \cup \mathcal{A}_2, \alpha_3 = \alpha_1 \oplus \alpha_2, \delta_3 = \delta_1 \oplus \delta_2, \text{ and} \\ (p_3, v_3) \in \mathcal{V}_3 \text{ implies, either } \mathcal{I}^{\mathcal{F}_3, \mathcal{A}_3, \alpha_3, \mathcal{D}, \delta_3, \mathtt{d}}_{(p_3, v_3)} \models_{\mathtt{d}} \psi_1 \vee \psi_2 \end{array} \right) \\
&\text{iff } \mathcal{F}_3 = \mathtt{ext}(\Psi_1 + \Psi_2).
\end{aligned}
$$
□

Note that, the next lemma can be reformulated for the logical version of the involved operators, namely for $\mathfrak{L}_{\mathrm{eql}}(X, Y, \ddot{\alpha}, \mathcal{D}, \ddot{\delta}, \mathtt{d})$. Anyway, this result is formalized in the Theorem 5.12.3.

Lemma 5.11 (The operators $\star$ and $+$ on $\mathfrak{A}_{\text{eql}}(X, Y, \ddot{\alpha}, \mathcal{D}, \ddot{\delta})$). *Let assume $\mathbb{A}_i$ be two AFMs $(X, Y, \ddot{\alpha}, \mathcal{D}, \ddot{\delta}, \mathcal{V}_i) \in \mathfrak{A}_{\text{eql}}(X, Y, \ddot{\alpha}, \mathcal{D}, \ddot{\delta})$ where $i = 1, 2$, we have that: $\mathbb{A}_1 \star \mathbb{A}_2 = \mathbb{A}_1 + \mathbb{A}_2 = (X, Y, \ddot{\alpha}, \mathcal{D}, \ddot{\delta}, \mathcal{V}_1 \cup \mathcal{V}_2)$.*

Proof. Straightforward from the definitions of $\star$ and $+$. □

Given $[\Psi_1]$, $[\Psi_2] \in [\mathfrak{L}_{\text{eql}}(X, Y, \ddot{\alpha}, \mathcal{D}, \ddot{\delta}, \mathtt{d})]$, we define (with an abuse of notation): $[\Psi_1] \leq [\Psi_2]$ as $\Psi_1 \leq \Psi_2$, $[\Psi_1] \bullet [\Psi_2] = [\Phi_1 \bullet \Psi_2]$, $[\Psi_1] \star [\Psi_2] = [\Psi_1 \star \Psi_2]$, $[\Psi_1] + [\Psi_2] = [\Psi_1 + \Psi_2]$, and $[\Psi_1]^\dagger = [\Psi_1{}^\dagger]$.

Theorem 5.12 (`ext` is a lattice monomorphism). *Let X be a set of features, let Y a set of attributes, let $\mathcal{D}$ be a set of domains, let $\ddot{\alpha}$ be a function of $Y \to X$, let $\ddot{\delta}$ be a function of $Y \to \mathcal{D}$ and let $\mathtt{d}$ is a function of $\mathcal{D} \to \cup\mathcal{D}$ such that $\mathtt{d} \propto \ddot{\delta}$.*

1. *$([\mathfrak{L}(X, Y, \ddot{\alpha}, \mathcal{D}, \ddot{\delta}, \mathtt{d})], \leq)$ is a bounded lattice with join $\bullet$, meet $\star$, bottom $[(\emptyset, \emptyset, \bot, \mathcal{D}, \bot, \mathtt{d}, \mathit{true})]$ and top $[(X, Y, \ddot{\alpha}, \mathcal{D}, \ddot{\delta}, \mathtt{d}, \mathit{false})]$. Moreover, the mapping `ext` is a bounded lattice monomorphism from $([\mathfrak{L}(X, Y, \ddot{\alpha}, \mathcal{D}, \ddot{\delta}, \mathtt{d})], \leq)$ to $(\mathfrak{A}(X, Y, \ddot{\alpha}, \mathcal{D}, \ddot{\delta}), \leq)$.*

2. *If X has infinitely many elements, then $[\mathfrak{L}_{\text{fin}}(X, Y, \ddot{\alpha}, \mathcal{D}, \ddot{\delta}, \mathtt{d})]$ is a sublattice of $[\mathfrak{L}(X, Y, \ddot{\alpha}, \mathcal{D}, \ddot{\delta}, \mathtt{d})]$ with the same bottom and no top. Moreover, `ext` is a lattice isomorphism from $[\mathfrak{L}_{\text{fin}}(X, Y, \ddot{\alpha}, \mathcal{D}, \ddot{\delta}, \mathtt{d})]$ to $\mathfrak{A}_{\text{fin}}(X, Y, \ddot{\alpha}, \mathcal{D}, \ddot{\delta})$.*

3. *$[\mathfrak{L}_{\text{eql}}(X, Y, \ddot{\alpha}, \mathcal{D}, \ddot{\delta}, \mathtt{d})]$ is a sublattice of $[\mathfrak{L}(X, Y, \ddot{\alpha}, \mathcal{D}, \ddot{\delta}, \mathtt{d})]$ and it is a Boolean lattice with bottom $[(X, Y, \ddot{\alpha}, \mathcal{D}, \ddot{\delta}, \mathtt{d}, \mathit{true})]$, same top of $[\mathfrak{L}(X, Y, \ddot{\alpha}, \mathcal{D}, \ddot{\delta}, \mathtt{d})]$, complement †, and where the meet behaves like $+$. Moreover, the mapping `ext` is a Boolean lattice monomorphism from $[\mathfrak{L}_{\text{eql}}(X, Y, \ddot{\alpha}, \mathcal{D}, \ddot{\delta}, \mathtt{d})]$ to $\mathfrak{A}_{\text{eql}}(X, Y, \ddot{\alpha}, \mathcal{D}, \ddot{\delta})$ and it is an isomorphism whenever X is finite.*

Proof. Straightforward from Theorems 3.16, 5.5-5.9 and Lemmas 3.12, 3.14, 5.10, 5.11. □

5.3. Logical Characterization of Slices and Interfaces for AFM

The following theorem provides a logical characterization of the slice operator for AFM.

Theorem 5.13 (Logical characterization of the operator Π_Y). *Let us assume $\Psi = (\mathcal{F}, \mathcal{A}, \alpha, \mathcal{D}, \delta, \mathtt{d}, \psi) \in \mathfrak{L}(X, Y, \ddot{\alpha}, \mathcal{D}, \ddot{\delta}, \mathtt{d})$, $\mathcal{F}' \subseteq X$ and $\mathcal{A}' \subseteq Y$. We define: $\Pi_{\mathcal{F}', \mathcal{A}'}(\Psi) = \big(\mathcal{F} \cap \mathcal{F}', \mathcal{A} \cap \mathcal{A}_R, \alpha|_{\mathcal{A}_R}, \mathcal{D}, \delta|_{\mathcal{A}_R}, \curlyvee_{\text{ftrs}(\psi^\dagger) \setminus \mathcal{F}'}(\psi^\dagger)\big)$ such that $\mathcal{A}_R = \alpha^{-1}(\mathcal{F}') \cap \mathcal{A}'$ and $\psi^\dagger$ denotes $(\overrightarrow{\exists a_i{:}D_i}.\psi)$ where $\text{attr}(\psi) \cap \mathcal{A}' = \{a_1, \ldots, a_n\}$ and $D_i = \delta(a_i)$. Then: $\Pi_{\mathcal{F}', \mathcal{A}'}(\mathit{ext}(\Psi)) = \mathit{ext}(\Pi_{\mathcal{F}', \mathcal{A}'}(\Psi))$.*

PROOF. Let $\mathtt{ext}(\Psi) = (\mathcal{F}, \mathcal{A}, \alpha, \mathcal{D}, \delta, \mathcal{V})$ where $\mathcal{V} = \{(p, v) \in 2^{\mathcal{F}} \times 2^{\alpha^{-1}(p) \to \cup \mathcal{D}} \mid v \propto \delta \text{ and } \mathcal{I}^{\Psi}_{(p,v)} \models_{\mathtt{d}} \psi\}$. We have:

$$
\begin{array}{l}
\Pi_{\mathcal{F}', \mathcal{A}'}(\mathtt{ext}(\Psi)) = (\mathcal{F}_3, \mathcal{A}_3, \alpha_3, \mathcal{D}, \delta_3, \mathcal{V}_3) \\
\text{iff} \left(\begin{array}{l} \mathcal{F}_3 = \mathcal{F} \cap \mathcal{F}', \mathcal{A}_3 = \mathcal{A} \cap \mathcal{A}_R, \alpha_3 = \alpha|_{\mathcal{A}_R}, \delta_3 = \delta|_{\mathcal{A}_R} \ , \\ \text{ and } \mathcal{V}_3 = \{(p|_{\mathcal{F}'}, v|_{\mathcal{A}_R}) \mid (p, v) \in \mathcal{V}\} \end{array} \right) \\
\text{iff} \left(\begin{array}{l} \mathcal{F}_3 = \mathcal{F} \cap \mathcal{F}', \mathcal{A}_3 = \mathcal{A} \cap \mathcal{A}_R, \alpha_3 = \alpha|_{\mathcal{A}_R}, \delta_3 = \delta|_{\mathcal{A}_R} \text{ and} \\ \mathcal{V}_3 = \{(p|_{\mathcal{F}'}, v|_{\mathcal{A}_R}) \mid \mathrm{dom}(v|_{\mathcal{A}_R}) = \alpha_3^{-1}(p|_{\mathcal{F}'}) \ , \ v \propto \delta \text{ and } \mathcal{I}^{\Psi}_{(p,v)} \models_{\mathtt{d}} \psi\} \end{array} \right) \\
\text{iff} \left(\begin{array}{l} \mathcal{F}_3 = \mathcal{F} \cap \mathcal{F}', \mathcal{A}_3 = \mathcal{A} \cap \mathcal{A}_R, \alpha_3 = \alpha|_{\mathcal{A}_R}, \delta_3 = \delta|_{\mathcal{A}_R} \text{ and} \\ \mathcal{V}_3 = \left\{ (p|_{\mathcal{F}'}, v|_{\alpha^{-1}(p)})|_{\mathcal{A}'}) \;\middle|\; \begin{array}{l} \mathrm{dom}(v|_{\alpha^{-1}(p)}) = \left(\alpha|_{\mathcal{A}'}\right)^{-1}(p|_{\mathcal{F}'}) \ , \\ v \propto \delta \text{ and } \mathcal{I}^{\Psi}_{(p, v|_{\mathcal{A}'})} \models_{\mathtt{d}} \overrightarrow{\exists a_i : D_i}.\psi \end{array} \right\} \end{array} \right) \\
\text{iff} \left(\begin{array}{l} \mathcal{F}_3 = \mathcal{F} \cap \mathcal{F}', \mathcal{A}_3 = \mathcal{A} \cap \mathcal{A}_R, \alpha_3 = \alpha|_{\mathcal{A}_R}, \delta_3 = \delta|_{\mathcal{A}_R} \text{ and} \\ \mathcal{V}_3 = \left\{ (p, v) \;\middle|\; \begin{array}{l} \mathrm{dom}(v) = \alpha^{-1}(p) \ , \ v \propto \delta \text{ and} \\ \mathcal{I}^{\mathcal{F}_3, \mathcal{A}_3, \alpha_3, \mathcal{D}, \delta_3, \mathtt{d}}_{(p,v)} \models_{\mathtt{d}} \curlyvee_{\mathrm{ftrs}(\psi^\dagger) \setminus \mathcal{F}'} \left(\overrightarrow{\exists a_i : D_i}.\psi\right) \end{array} \right\} \end{array} \right) \\
\text{iff } \mathtt{ext}(\Pi_{\mathcal{F}', \mathcal{A}'}(\Psi)) = (\mathcal{F}_3, \mathcal{A}_3, \alpha_3, \mathcal{D}, \delta_3, \mathcal{V}_3). \qquad \square
\end{array}
$$

The following corollary provides a logical characterization of the interface relation $\mathbb{A}_1 \preceq \mathbb{A}_2$ which is the same as the interpretation of the slice operator $\mathbb{A}_1 = \Pi_{\mathcal{F}', \mathcal{A}'}(\mathbb{F}_2)$ when $\mathcal{F}'$ and $\mathcal{A}'$ are the features and the attributes of $\mathbb{A}_1$ respectively (cf. Theorem 5.13).

Corollary 5.14 (Logical characterization of the relation $\preceq$). *Let assume us Ψ_i be two AFMs $(\mathcal{F}_i, \mathcal{A}_i, \alpha_i, \mathcal{D}, \delta_i, \mathtt{d}_i, \psi_i) \in \mathfrak{L}(X, Y, \ddot{\alpha}, \mathcal{D}, \ddot{\delta}, \mathtt{d})$ where $i = 1, 2$. We write $\Psi_1 \preceq \Psi_2$ to mean that $\mathcal{F}_1 \subseteq \mathcal{F}_2$, $\mathcal{A}_1 \subseteq \mathcal{A}_2$, $\alpha_1 \subseteq \alpha_2$, $\delta_1 \subseteq \delta_2$ and $\psi_1 \equiv_{\mathtt{d}} \curlyvee_{\mathrm{ftrs}(\psi^\dagger) \setminus \mathcal{F}_1} \left(\psi^\dagger\right)$ where $\psi^\dagger$ denotes $(\overrightarrow{\exists a_i : D_i}.\psi_2)$ where $\{a_1, \ldots, a_n\} = (\mathcal{A}_2 \setminus \mathcal{A}_1) \cap \mathrm{attr}(\psi_2)$ and $D_i = \delta_2(a_i)$. Then: $\mathtt{ext}(\Psi_1) \preceq \mathtt{ext}(\Psi_2)$ holds if and only $\Psi_1 \preceq \Psi_2$ holds.*

PROOF. Let $\mathtt{ext}(\Psi_i) = (\mathcal{F}_i, \mathcal{A}_i, \alpha_i, \mathcal{D}, \delta_i, \mathcal{V}_i)$ where $i = 1, 2$ such that $\mathcal{V}_i = \{(p_i, v_i) \in 2^{\mathcal{F}_i} \times 2^{\alpha_i^{-1}(p_i) \to \cup \mathcal{D}} \mid v_i \propto \delta_i \text{ and } \mathcal{I}^{\Psi_i}_{(p_i, v_i)} \models_{\mathtt{d}} \psi_i\}$.

$$
\begin{array}{l}
\text{We have: } \mathtt{ext}(\Psi_1) \preceq \mathtt{ext}(\Psi_2) \\
\text{iff} \left(\begin{array}{l} \mathcal{F}_1 \subseteq \mathcal{F}_2, \mathcal{A}_1 \subseteq \mathcal{A}_2, \alpha_1 \subseteq \alpha_2, \delta_1 \subseteq \delta_2, \\ \mathcal{V}_1 = \{(p_2|_{\mathcal{F}_1}, v_2|_{\mathcal{A}_1}) \mid (p_2, v_2) \in \mathcal{V}_2\} \end{array} \right) \text{ by Definition 3.25 ,} \\
\text{iff} \left(\begin{array}{l} \mathcal{F}_1 \subseteq \mathcal{F}_2, \mathcal{A}_1 \subseteq \mathcal{A}_2, \alpha_1 \subseteq \alpha_2, \delta_1 \subseteq \delta_2 \\ \{(p_1, v_1) \mid \mathcal{I}^{\Psi_1}_{(p_1, v_1)} \models_{\mathtt{d}} \psi_1\} = \{(p_2|_{\mathcal{F}_1}, v_2|_{\mathcal{A}_1}) \mid \mathcal{I}^{\Psi_2}_{(p_2, v_2)} \models_{\mathtt{d}} \psi_2\} \end{array} \right) \\
\text{iff} \left(\begin{array}{l} \mathcal{F}_1 \subseteq \mathcal{F}_2, \mathcal{A}_1 \subseteq \mathcal{A}_2, \alpha_1 \subseteq \alpha_2, \delta_1 \subseteq \delta_2, \\ \mathcal{I}^{\Psi_1}_{(p_2|_{\mathcal{F}_1}, v_2|_{\mathcal{A}_1})} \models_{\mathtt{d}} \psi_1 \text{ iff } \mathcal{I}^{\Psi_1}_{(p_2|_{\mathcal{F}_1}, v_2|_{\mathcal{A}_1})} \models_{\mathtt{d}} \curlyvee_{\mathrm{ftrs}(\psi^\dagger) \setminus \mathcal{F}_1} \left(\psi^\dagger\right) \end{array} \right) \\
\text{iff } \mathcal{F}_1 \subseteq \mathcal{F}_2, \mathcal{A}_1 \subseteq \mathcal{A}_2, \alpha_1 \subseteq \alpha_2, \delta_1 \subseteq \delta_2, \text{ and } \psi_1 \equiv_{\mathtt{d}} \curlyvee_{\mathrm{ftrs}(\psi^\dagger) \setminus \mathcal{F}_1} \left(\psi^\dagger\right) \\
\text{iff } \Psi_1 \leq \Psi_2. \qquad \square
\end{array}
$$

6. Related Work

Although, in the literature, the logical representation of both FMs (see, e.g., Sect. 2.3 of Apel *et al.* [1]) and AFMs [22, 23, 12, 24] are well known, we are not

aware of any work that (as done in the present paper) provides a formal account of the correspondence between the algebraic and the logical characterizations operators and relations for FMs and AFMs.

The investigation presented in this paper started from the FM composition operator $\bullet$ and the induced fragment partial order relation $\leq$. In the following we briefly discuss relevant related work on FM composition operators and on FM relations.

Composition operators for FMs and AFMs are often investigated in connection with multi software product lines, which are sets of interdependent product lines [38]. Eichelberger and Schmid [39] present an overview of textual-modeling languages which support variability-model composition for FMs (like FAMILIAR [40]) and AFMs (like VELVET [9], TVL [12], VSL [41]) and discuss their support for composition, modularity, and evolution. Acher *et al.* [11] consider different FM composition operators together with possible implementations and discuss advantages and drawbacks.

The FM fragment relation introduced in this paper generalizes the FM interface relation introduced by Schröter *et al.* [18], which (see Remark 3.19) is closely related to the FM slice opeator introduced by Acher *et al.* [36]. The work of Acher *et al.* [36] focuses on FM decomposition. In subsequent work [42], Acher *et al.* use the slice operator in combination with a merge operator to address evolutionary changes for extracted variability models, focusing on detecting differences between feature-model versions during evolution. Analyzing fragmented feature models usually requires to compose the fragments in order to apply existing techniques [17, 43]. Schröter *et al.* [18] proposed feature model interfaces to support evolution of large FMs composed by several FMs fragments. Namely, they propose to analyze a fragmented FM where some fragments have been replaced by carefully chosen FM interface to obtain results that hold for the original FM and for all its evolution where the evolved version of the fragments replaced by the interfaces are still compatible with the interfaces. More recently, Lienhardt *et al.* [19] strengthen FM interfaces to support efficient automated product discovery in fragmented FMs. We are not aware of other works investigating similar order relations between AFMs.

7. Conclusion and Future Work

The formalization presented in this paper sheds new light on the correspondence between the algebraic and logical characterizations of operations and relations for FMs and AFMs. Namely, it connects the two characterizations by monomorphisms from lattices of logical FMs and AFMs to lattices of extensional FMs and AFMs, respectively. It aims to foster the development of a formal framework for supporting practical exploitation of future theoretical developments on FMs, AFMs and (multi) software product lines. For instance, recent works [18, 19] which introduced novel FM relations by relying on the extensional representation for theory and on the logical representation for experiments, do not show the logical representation of the relations.

In future work we would like to extend this picture by considering other FM and AFM representations [2, 1], operators [11] and relations [19]. We are also planning to adapt out formalization to cardinality-based FMs [44]. Moreover, we want to investigate how the fragment relation could be exploited, in real use cases, to decompose large FMs and AFMs in manageable parts. Recently [34], we have introduced the notion of software product line signature in order to express dependencies between different product lines, and we have lifted to software product lines the notions of FM composition and interface. In future work we would like to lift to software product lines other FM operations and relations and to provide a formal account of the connection between different software product line implementation approaches [45, 17, 1]. This formalization would enable formal reasoning on multi software product lines comprising software product lines implemented according to different approaches. Some delta-oriented programming languages for software product lines of Java-like programs, like ABS [46] and Parametric DeltaJ [29], support propagating feature attributes to the deltas (and therefore to the generated variants). The aforementioned notion of software product line signature [34], which has been introduced to provide a formal foundation for delta-oriented multi software product lines of Java-like programs, does not consider feature attributes. In future work we would like to extend it to consider feature attributes.

Acknowledgments. We thank the anonymous ICTAC 2020 and TCS reviewers for their comments and suggestions.

References

[1] S. Apel, D. S. Batory, C. Kästner, G. Saake, Feature-Oriented Software Product Lines: Concepts and Implementation, Springer, 2013. doi:10.1007/978-3-642-37521-7.

[2] D. Batory, Feature models, grammars, and propositional formulas, in: Proceedings of International Software Product Line Conference (SPLC), Vol. 3714 of LNCS, Springer, 2005, pp. 7–20. doi:10.1007/11554844_3.

[3] K. C. Kang, S. G. Cohen, J. A. Hess, W. E. Novak, A. S. Peterson, Feature-Oriented Domain Analysis (FODA) Feasibility Study, Tech. Rep. CMU/SEI-90-TR-21, Carnegie Mellon Software Engineering Institute (1990).

[4] K. Czarnecki, T. Bednasch, P. Unger, U. Eisenecker, Generative programming for embedded software: An industrial experience report, in: D. Batory, C. Consel, W. Taha (Eds.), Generative Programming and Component Engineering, Springer Berlin Heidelberg, Berlin, Heidelberg, 2002, pp. 156–172.

[5] T. Berger, S. She, R. Lotufo, A. Wąsowski, K. Czarnecki, Variability modeling in the real: a perspective from the operating systems domain, in:

Proc. 25th International Conference on Automated Software Engineering (ASE 2010), ACM Press, 2010, pp. 73–82. doi:0.1145/1858996.1859010.

[6] T. Berger, R. Rublack, D. Nair, J. M. Atlee, M. Becker, K. Czarnecki, A. Wąsowski, A survey of variability modeling in industrial practice, in: Proc. 7th International Workshop on Variability Modelling of Software-Intensive Systems, ACM Press, 2013, pp. 7:1–7:8.

[7] R. Tartler, D. Lohmann, J. Sincero, W. Schröder-Preikschat, Feature consistency in compile-time-configurable system software: facing the linux 10,000 feature problem, in: Proc. 6th European Conference on Computer systems (EuroSys 2011), ACM Press, 2011, pp. 47–60. doi:10.1145/1966445.1966451.

[8] M. Lienhardt, F. Damiani, S. Donetti, L. Paolini, Multi software product lines in the wild, in: Proc. 12th International Workshop on Variability Modelling of Software-Intensive Systems, VAMOS 2018, ACM, 2018, pp. 89–96. doi:10.1145/3168365.3170425.

[9] M. Rosenmüller, N. Siegmund, T. Thüm, G. Saake, Multi-dimensional variability modeling, in: Proc. 5th International Workshop on Variability Modelling of Software-Intensive Systems, ACM Press, 2011, pp. 11–20. doi:10.1145/1944892.1944894.

[10] M. Acher, P. Collet, P. Lahire, R. B. France, Comparing approaches to implement feature model composition, in: Proc. 6th European Conference on Modelling Foundations and Applications (ECMFA 2010), Springer, 2010, pp. 3–19.

[11] M. Acher, B. Combemale, P. Collet, O. Barais, P. Lahire, R. B. France, Composing your compositions of variability models, in: Proc. 16th International Conference on Model-Driven Engineering Languages and Systems (MODELS 2013), Springer, 2013, pp. 352–369. doi:10.1007/978-3-642-41533-3_22.

[12] A. Classen, Q. Boucher, P. Heymans, A text-based approach to feature modelling: Syntax and semantics of TVL, Science of Computer Programming 76 (12) (2011) 1130 – 1143. doi:10.1016/j.scico.2010.10.005.

[13] M. Rosenmüller, N. Siegmund, C. Kästner, S. S. U. Rahman, Modeling dependent software product lines, in: Proc. Workshop on Modularization, Composition and Generative Techniques for Product Line Engineering, 2008, pp. 13–18.

[14] R. Schröter, T. Thüm, N. Siegmund, G. Saake, Automated analysis of dependent feature models, in: Proc. 7th International Workshop on Variability Modelling of Software-Intensive Systems, ACM Press, 2013, pp. 9:1–9:5. doi:10.1145/2430502.2430515.

[15] M. Mendonca, A. Wasowski, K. Czarnecki, SAT-based analysis of feature models is easy, in: D. Muthig, J. D. McGregor (Eds.), Proceedings of the 13th International Software Product Line Conference, Vol. 446 of ACM International Conference Proceeding Series, ACM, 2009, pp. 231–240. doi:10.5555/1753235.1753267.

[16] D. Benavides, S. Segura, A. Ruiz-Cortés, Automated analysis of feature models 20 years later: A literature review, Information Systems 35 (6) (2010) 615–636. doi:10.1016/j.is.2010.01.001.

[17] T. Thüm, S. Apel, C. Kästner, I. Schaefer, G. Saake, A classification and survey of analysis strategies for software product lines, ACM Comput. Surv. 47 (1) (2014) 6:1–6:45. doi:10.1145/2580950.

[18] R. Schröter, S. Krieter, T. Thüm, F. Benduhn, G. Saake, Feature-model interfaces: The highway to compositional analyses of highly-configurable systems, in: Proceedings of the 38th International Conference on Software Engineering, ICSE '16, ACM, 2016, pp. 667–678. doi:10.1145/2884781.2884823.

[19] M. Lienhardt, F. Damiani, E. B. Johnsen, J. Mauro, Lazy product discovery in huge configuration spaces, in: Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering, ICSE '20, Association for Computing Machinery, New York, NY, USA, 2020, p. 1509–1521. doi:10.1145/3377811.3380372.

[20] G. Foundation, Gentoo linux, last visited, 2019-08-20 (2019). URL `https://gentoo.org`

[21] F. Damiani, M. Lienhardt, L. Paolini, On two characterizations of feature models, in: 17th International Colloquium on Theoretical Aspects of Computing (ICTAC), Vol. 12545 of Lecture Notes in Computer Science, Springer, Berlin, Germany, 2020, pp. 1–21. doi:10.1007/978-3-030-64276-1_6.

[22] D. Benavides, P. Trinidad, A. Ruiz-Cortés, Automated reasoning on feature models, in: O. Pastor, J. Falcão e Cunha (Eds.), Advanced Information Systems Engineering, Springer Berlin Heidelberg, Berlin, Heidelberg, 2005, pp. 491–503.

[23] A. S. Karataş, H. Oğuztüzün, A. Doğru, From extended feature models to constraint logic programming, Science of Computer Programming 78 (12) (2013) 2295–2312, special Section on International Software Product Line Conference 2010 and Fundamentals of Software Engineering (selected papers of FSEN 2011). doi:https://doi.org/10.1016/j.scico.2012.06.004.

[24] U. Lesta, I. Schaefer, T. Winkelmann, Detecting and explaining conflicts in attributed feature models, in: J. M. Atlee, S. Gnesi (Eds.), Proceedings 6th Workshop on Formal Methods and Analysis in SPL Engineering,

FMSPLE@ETAPS 2015, London, UK, 11 April 2015, Vol. 182 of EPTCS, 2015, pp. 31–43. doi:10.4204/EPTCS.182.3.

[25] E. P. K. Tsang, Foundations of constraint satisfaction., Computation in cognitive science, Academic Press, 1993.

[26] J. Petke, Bridging Constraint Satisfaction and Boolean Satisfiability, Artificial Intelligence: Foundations, Theory, and Algorithms, Springer, 2015.

[27] G. Bécan, R. Behjati, A. Gotlieb, M. Acher, Synthesis of attributed feature models from product descriptions, in: Proceedings of the 19th International Conference on Software Product Line, SPLC '15, Association for Computing Machinery, New York, NY, USA, 2015, p. 1–10. doi:10.1145/2791060.2791068.

[28] A. S. Karataş, H. Oğuztüzün, Attribute-based variability in feature models, Requir. Eng. 21 (2) (2016) 185–208. doi:10.1007/s00766-014-0216-9.

[29] T. Winkelmann, J. Koscielny, C. Seidl, S. Schuster, F. Damiani, I. Schaefer, Parametric deltaj 1.5: Propagating feature attributes into implementation artifacts, in: Gemeinsamer Tagungsband der Workshops der Tagung Software Engineering 2016 (SE 2016), Wien, 23.-26. Februar 2016, Vol. 1559 of CEUR Workshop Proceedings, CEUR-WS.org, 2016, pp. 40–54.
URL `http://ceur-ws.org/Vol-1559/paper04.pdf`

[30] F. Roos-Frantz, D. Benavides, A. R. Cortés, A. Heuer, K. Lauenroth, Quality-aware analysis in product line engineering with the orthogonal variability model, Softw. Qual. J. 20 (3-4) (2012) 519–565. doi:10.1007/s11219-011-9156-5.

[31] R. Lotufo, S. She, T. Berger, K. Czarnecki, A. Wąsowski, Evolution of the linux kernel variability model, in: Proceedings of the 14th International Conference on Software Product Lines: Going Beyond, SPLC'10, Springer-Verlag, Berlin, Heidelberg, 2010, pp. 136–150. doi:10.1007/978-3-642-15579-6_10.

[32] E. F. Codd, A relational model of data for large shared data banks, Commun. ACM 13 (6) (1970) 377–387. doi:10.1145/362384.362685.

[33] F. Damiani, M. Lienhardt, L. Paolini, A formal model for multi SPLs, in: 7th International Conference on Fundamentals of Software Engineering (FSEN), Vol. 10522 of Lecture Notes in Computer Science, Springer, Berlin, Germany, 2017, pp. 67–83. doi:10.1007/978-3-319-68972-2_5.

[34] F. Damiani, M. Lienhardt, L. Paolini, A formal model for multi software product lines, Science of Computer Programming 172 (2019) 203 – 231. doi:10.1016/j.scico.2018.11.005.

[35] B. A. Davey, H. A. Priestley, Introduction to Lattices and Order, 2nd Edition, Cambridge University Press, 2002. doi:10.1017/CBO9780511809088.

[36] M. Acher, P. Collet, P. Lahire, R. B. France, Slicing feature models, in: 26th IEEE/ACM International Conference on Automated Software Engineering, (ASE), 2011, 2011, pp. 424–427. doi:10.1109/ASE.2011.6100089.

[37] M. Ben-Ari, Mathematical Logic for Computer Science, 3rd Edition, Springer Publishing Company, Incorporated, 2012.

[38] G. Holl, P. Grünbacher, R. Rabiser, A systematic review and an expert survey on capabilities supporting multi product lines, Information & Software Technology 54 (8) (2012) 828–852. doi:10.1016/j.infsof.2012.02.002.

[39] H. Eichelberger, K. Schmid, A systematic analysis of textual variability modeling languages, in: Proc. 17th International Software Product Line Conference (SPLC 2013), ACM Press, 2013, pp. 12–21. doi:10.1145/2491627.2491652.

[40] M. Acher, P. Collet, P. Lahire, R. B. France, Familiar: A domain-specific language for large scale management of feature models, Science of Computer Programming 78 (6) (2013) 657–681. doi:10.1016/j.scico.2012.12.004.

[41] A. Abele, Y. Papadopoulos, D. Servat, M. Törngren, M. Weber, The CVM framework - A prototype tool for compositional variability management, in: Proc. 4th International Workshop on Variability Modelling of Software-Intensive Systems, Vol. 37 of ICB-Research Report, Universität Duisburg-Essen, 2010, pp. 101–105.

[42] M. Acher, A. Cleve, P. Collet, P. Merle, L. Duchien, P. Lahire, Extraction and evolution of architectural variability models in plugin-based systems, Software and Systems Modeling 13 (4) (2014) 1367–1394. doi:10.1007/s10270-013-0364-2.

[43] J. A. Galindo, D. Benavides, P. Trinidad, A. M. Gutiérrez-Fernández, A. Ruiz-Cortés, Automated analysis of feature models: Quo vadis?, Computing 101 (5) (2019) 387–433. doi:10.1007/s00607-018-0646-1.

[44] K. Czarnecki, S. Helsen, U. Eisenecker, Formalizing cardinality-based feature models and their specialization, Software Process: Improvement and Practice 10 (1) (2005) 7–29. doi:10.1002/spip.213.

[45] I. Schaefer, R. Rabiser, D. Clarke, L. Bettini, D. Benavides, G. Botterweck, A. Pathak, S. Trujillo, K. Villela, Software diversity: state of the art and perspectives, International Journal on Software Tools for Technology Transfer 14 (5) (2012) 477–495. doi:10.1007/s10009-012-0253-y.

[46] D. Clarke, R. Muschevici, J. Proença, I. Schaefer, R. Schlatte, Variability modelling in the ABS language, in: B. K. Aichernig, F. S. de Boer, M. M. Bonsangue (Eds.), Formal Methods for Components and Objects, Springer Berlin Heidelberg, Berlin, Heidelberg, 2011, pp. 204–224. doi:10.1007/978-3-642-25271-6_11.