Mining Complex Feature Correlations from Large Software Product Line Configurations

Bo Zhang
Software Engineering Research Group
University of Kaiserslautern
Kaiserslautern, Germany
bo.zhang@cs.uni-kl.de

***Abstract*— As a Software Product Line (SPL) evolves with increasing number of features and feature values, the feature correlations become extremely intricate, and the specifications of these correlations tend to be either incomplete or inconsistent with their realizations, causing misconfigurations in practice. In order to guide product configuration processes, we present a solution framework to recover complex feature correlations from existing product configurations. These correlations are further pruned automatically and validated by domain experts. During implementation, we use association mining techniques to automatically extract strong association rules as potential feature correlations. This approach is evaluated using a large-scale industrial SPL in the embedded system domain, and finally we identify a large number of complex feature correlations.**

***Index Terms*— Feature Correlation, Product Line Configuration, Association Mining.**

I. INTRODUCTION

Nowadays the development of Software Product Lines (SPLs) is often conducted in an incremental way [17], in which there are an increasing number of features and feature values with their correlations included in the SPL [8]. A feature correlation is a logical implication between of features or feature values based on domain knowledge. Feature correlations are often documented in a feature model [10] to facilitate SPL development and product configuration. They can indicate feature type (e.g., "mandatory" and "optional"), feature groups (e.g., "OR" and "XOR"), and cross-tree constraints (e.g., "requires" and "excludes") [5][6]. A basic feature correlation could be simply a pairwise dependency between two features, e.g., $F1 \rightarrow F2$. However, since a feature is often selected by different products and is assigned different feature values, the feature correlations in practice are usually in an more intricate form, involving two sets of multiple feature assignments, e.g., $(F1=10 \wedge F2=0xFF) \rightarrow (F3=True \wedge F4=$ "EUR").

Moreover, as SPL specifications and realizations evolve both in space and in time during evolution [11][14][20], the feature correlations are changing over time. As a result, the specifications of these correlations tend to be either incomplete or inconsistent with their realizations. For instance, a feature correlation is added in the feature model but not updated in the code, or vice versa. This is a practical problem in the product configuration process during SPL application engineering, and it causes misconfigurations due to missing or incorrect feature correlations in SPL specifications [8][16].

Given this practical problem, we present a solution framework to recover complex feature correlations from existing product configurations. Each correlation involves two sets of multiple feature assignments, known as antecedent and concequent. Since the feature correlations are so intricate that it is very difficult to identify the correlation manually, we propose to use association mining techniques to automatically extract strong association rules of feature assignments. As one of the important applications of data mining, association rule mining [4] lends itself to the discovery of certain patterns from an existing dataset. The objective of association mining is the elicitation of useful or interesting rules from which new knowledge can be derived [9].

TABLE 1. An Example Configuration Matrix

	F1	F2	F3
P1	defined	EUR	512
P2	defined	EUR	
P3	defined	EUR	512
P4		USD	512

In our context of SPL development, we consider the configurations of all existing products as a dataset for association mining. For instance, Table 1 shows an example of a configuration dataset, called configuration matrix, that consists of four features (shown in columns) and their assigned values (if given) in three products (shown in rows). Using association mining techniques, we calculate strong association rules that satisfy a specified threshold of minimum support and minimum confidence. These association rules are considered as potential feature correlations. In order to reduce the number of these feature correlations, we further conduct an automatic correlation pruning by removing sub-rules that do not provide any predictive advantage. Finally the remaining feature correlations are validated by domain experts. The validated feature correlations can be used as prediction knowledge to

provide recommendations on selected features and assigned feature values during configurations of new products.

In our previous study [23], the correlation mining framework was presented and demonstrated with an industrial example. However, there was an scalability problem in the implementation of a third party association mining tool called Orange [15], and thus we restricted the input dataset and the derived association rules were incomplete. In this paper, we calculate frequent itemsets using a more advanced algorithm called LCM [21], and then we conduct association rule generation and pruning to derive complete and concise feature correlations with tool support.

This paper is organized as follows. The solution framework is presented in section 2 as an overview of our approach, and then each process of the framework is introduced in following sections. As the first step, configuration extraction is presented in section 3, and then the process of data preparation is discussed in section 4. While correlation mining is presented in section 5, the subsequent process of correlation pruning is introduced in section 6. Related studies are discussed in section 7, and finally the conclusion is presented in section 8.

II. SOLUTION FRAMEWORK

Given the problem of the implicit feature correlations among product configurations, we present a solution framework including a series of processes as shown in Fig. 1. The input of our approach is product configurations separately documented in each product. The first process is configuration extraction which analyzes all existing product configurations and results into a configuration matrix consisting of selected features with their values (if given) across all products (see Table 1). Then the second process of data preparation adapts the information in the configuration matrix by unifying the data format and discretizing continuous feature values. After that the configuration matrix is transformed into a dataset that is suitable for correlation mining.

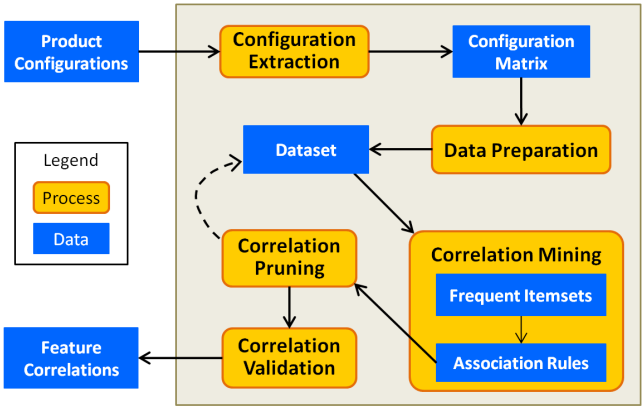


Figure 1. Feature Correlation Mining Framework.

During the correlation mining process, we use association mining techniques to identify significant association rules as potential feature correlations in two steps. In the first step, frequent itemsets of feature assignments are calculated that satisfy a specified threshold of minimum support. Then these frequent itemsets are used to identify strong association rules between two sets of feature assignments. Each association rule should satisfy a specified threshold of minimum confidence.

After correlation mining, the identified association rules have to be pruned because many rules are sub-rules and each of them can be implied by a parent rule with equal or smaller confidence. Therefore, these sub-rules do not provide any predictive advantage and should be removed from the set of feature correlations. Finally, each feature correlation is checked manually by domain experts to decide whether there exists a semantic relationship or that is only a coincidence.

Besides, the automated processes of correlation mining and pruning might be conducted repeatedly for different reasons. One scenario is that due to the large size of dataset the frequent itemsets and association rules can be generated incrementally. Another scenarios is that if the derived association rules are too less or too many, then the values of minimum support and minimum confidence might need to be adjusted by domain experts in order to shrink or extend the scope of association rules.

Our correlation mining approach is evaluated using an example of a large-scale industrial SPL in the embedded system domain. This industrial SPL contains 101 products, and each product has a separate configuration file written in XML language. In the following sections of this paper, we present the processes from configuration extraction to correlation pruning along with the industrial example. Each process is conducted automatically. The manual process of correlation validation is still in research progress and will not be introduced in this paper.

III. CONFIGURATION EXTRACTION

In this paper, product configurations are provided as the input for mining feature correlations. A product configuration includes selected features for this product, and sometimes a selected feature can be assigned a certain value. We extract such feature information from configurations of all existing products and build a configuration matrix of the SPL. Besides, we conduct various quantitative measurements in order to investigate the problem domain and guide our correlation mining process.

```
<Product name="P1">
    <Item name="F1"/>
    <Item name="F2">
        <Sel val="EUR"/>
    </Item>
    <Item name="F3">
        <Sel val="512"/>
    </Item>
</Product>
```

(a) An XML File

```
1 // Product P1
2 #define F1
3 #define F2  "EUR"
4 #define F3  512
5
6 //P1 Implementation
7 ...
```

(b) Macro Definitions

Figure 2. Product Configurations in Different Forms.

We notice that product configurations may exist in different forms in real SPLs. For instance, Fig. 2 shows the same product configuration in the form of an XML file and macro definition in source code respectively. Hence, different parsing techniques are needed to extract selected features and their

assigned values. In our previous work [22], we parsed configurations in macro definitions. However, since configurations in our industrial example are written in XML files, we use Python scripts to parse the XML configuration files and extract features and their assigned values. Moreover, since configurations are separated in the implementation of each product, it is necessary to synthesize the extracted feature information in order to identify their correlations. Therefore, we propose to build a configuration matrix as shown in Table 1, which is actually a feature-product table that documents selected features and their assigned values across all products.

Besides, if a feature is only selected without being assigned any value in all product configurations, then it is a binary feature. Otherwise it is a non-binary feature. A binary feature in the configuration matrix is assigned an artificial value "defined", such as **F**1 in Table 1. After parsing all the 101 configuration files of our industrial example, we build a configuration matrix with 100 valid products and 480 features (one configuration file is empty). Although it is not possible to present such a huge matrix in this paper, we conduct further measurements on the configuration matrix to investigate the problem domain.

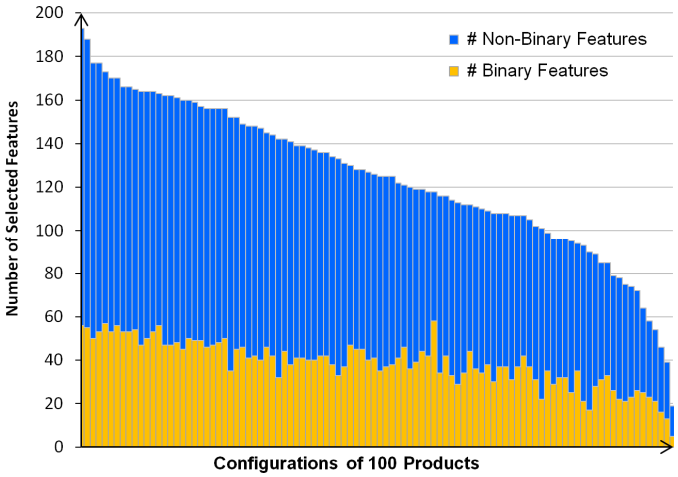
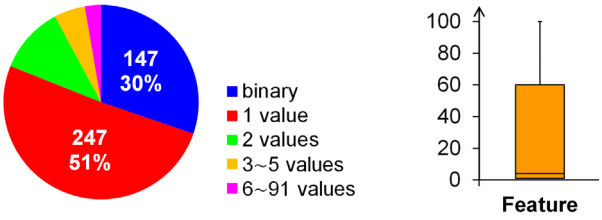


Figure 3. Number of Selected Features in Each Product.

Fig. 3 depicts the number of selected features in each product. The vertical axis indicates the numbers of binary and non-binary features in one product, while the horizontal axis indicates each corresponding product configuration and it is sorted by the total feature number of the product. We compare the number of binary features (yellow) and the number of non-binary features (blue). Finally each product has at least 19 features (rightmost) and at most 193 features (leftmost), and it has 124.7 features on average including 38.8 binary features and 85.9 non-binary features. It also shows that the number of non-binary features is always larger than the number of binary features across all products. Therefore, it is necessary to consider not only the correlations between features, but also the correlations between feature assignments.

Besides measuring the number of features in each product, we also focus on feature characterization across all products. The pie chart in Fig. 4(a) shows that there are 147 binary features (30%) and 247 features with only one value (51%). The remaining 86 features have at least two up to 91 values. Note that the features with one value should not be considered as binary features. For instance, **F**3 in Table 1 has been assigned only one value "512", but it could be assigned a different value in future product configurations.



(a) Feature Values (b) # Selecting Products

Figure 4. Feature Characterization across All Products.

Furthermore, the box plot in Fig. 4(b) shows that each feature is selected by at least one product and at most 100 products (which means it is a mandatory feature), and most features are selected by around four products (median value). The measurements shown in Fig. 4 indicate that on average each feature is selected by numerous products with different values, while many products have the same feature with the same value. Therefore, we assume that there are intensive correlations between feature assignments.

IV. DATA PREPARATION

In order to use the extracted configuration matrix for feature correlation mining, it is necessary to adapt the data in the matrix and make it suitable as a dataset for correlation mining algorithms. A dataset is formalized as follows. Let $\mathbb{F}$ = **F**1, **F**2, ..., **F**m be a set of features, and let $\mathbb{P}$ = **P**1, **P**2, ..., **P**n be a set of product configurations known as transactions in data mining. Each transaction of product **P** is a represented as a vector with the length of m, where **P**[i] = v if feature **F**i is selected by product **P** with the value v.

The conducted data adaptations are as follows.

A. Assigning Feature Values

During configuration extraction, a defined binary feature is assigned an artificial value "defined", because the dataset needs to have a consistent scheme for correlation mining. However, there might be absent features which are not defined in a corresponding product, and it is unclear whether there exists an exclusive correlation between an absent feature and the other defined features in the product. If such exclusive correlation can be confirmed by domain experts, then our approach can be also applied to mine exclusive feature correlations (e.g., **F**1="defined"↔**F**2="USD"). In this paper, we only focus on mining implicative correlations (e.g., **F**1="defined"⟶**F**3=512).

B. Encoding Feature Assignments

Given the large dataset containing various features and feature values, we encode each feature assignment (i.e., a pair of feature name and value) in the configuration matrix into a unique identifier, so that the encoded dataset can be used to conduct correlation mining and pruning in an efficient way. Therefore, each transaction in the encoded dataset is simply a set of identifiers instead of feature assignments. For instance,

the configuration matrix in Table 1 is encoded as shown in Table 2. In our industrial example, totally 742 different feature assignments are encoded into numeric identifiers.

TABLE 2. An Encoded Dataset

P1	item1	item2	item4
P2	item1	item2	
P3	item1	item2	item4
P4		item3	item4

V. CORRELATION MINING

In this section, we use association mining techniques [4] to identify correlations between feature assignments from a dataset of the configuration matrix. The correlation mining process is conducted in two steps. The first step is to calculate frequent features itemsets, while the second step is to construct feature association rules by splitting a frequent itemset into the antecedent and consequent of a rule. Due to the absence of domain knowledge, we set the association threshold of minimum support and minimum confidence to be very high (both 0.95). Finally the association rules are considered as potential feature correlations.

A. Frequent Itemsets

In order to derive association rules from a dataset, it is necessary to first calculate frequent itemsets. A frequent itemset is a set of items that exists with at least a specified percentage (called support) across all transactions (products). In our SPL context, each itemset is a set of feature assignments, and the support of an itemset is defined as the number of products containing these feature assignments divided by the number of all products, i.e.,

$$\text{Supp}(\boldsymbol{\Gamma}) = |\{\mathbf{P} \mid \boldsymbol{\Gamma} \text{ is selected in } \mathbf{P}\}| \,/\, |\mathbb{P}|$$

where $\boldsymbol{\Gamma}$ is an itemset of encoded feature assignments, and $\mathbf{P}$ is a product in all products $\mathbb{P}$. For instance, in Table 1 the support of the itemset {item1, item5} (i.e., {**F**1=defined, **F**3=512}) is 2/3. An itemset is frequent if its support is equal or larger than a minimum value *minsup*, which is usually larger than 0.5 and given by domain experts.

Agrawal and Srikant [2] observed that any subset of a frequent itemset is also frequent, known as the *downward closure* property. Based on this property, they developed the classic Apriori algorithm to calculate all frequent itemsets. The algorithm starts from an empty set by adding items one by one, and builds a frequent itemset of size k iteratively from its subset of size k-1. However, in our previous study [23] the calculation of frequent itemsets with all 480 features across 100 products using this algorithm was time-consuming and finally leads to a memory overflow. In this paper, in order to derive complete feature correlations we use a more advanced algorithm LCM [21] to calculate frequent itemsets. The algorithm is also implemented as an open source tool [12].

In our industrial example, we set the *minsup* to be 0.95 in the LCM tool and get 4029 frequent itemsets from 480 features across 100 products. We calculate frequent itemsets with at least two items, because itemsets with single items cannot be used to generate association rules. The bubble chart in Fig. 5 illustrates a distribution of these itemsets in terms of support and size. The support value ranges from 0.95 to 1.00 indicating their frequency of occurrence, while the size of these itemsets ranges from two to Max. ten. The largest bubble indicates a cluster of 649 frequent itemsets with a support of 0.95 and five features. The chart indicates that most itemsets contain multiple features (5.14 on average) with strong support. Since the size of a frequent itemset implies the size of an association rule derived from it, we expect to identify numerous complex association rules containing multiple features.

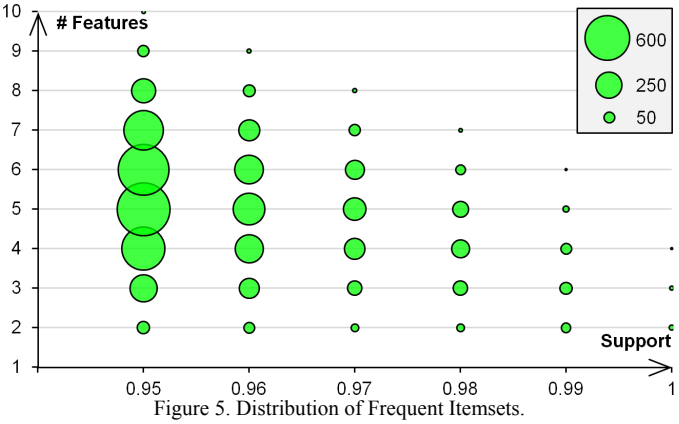


Figure 5. Distribution of Frequent Itemsets.

B. Association Rules

Based on the calculated frequent itemsets, the second step is to identify association rules that satisfy given thresholds of minimum support and minimum confidence. An association rule $\boldsymbol{\Upsilon}$ is an implication of the form $\langle \boldsymbol{A} \rightarrow \boldsymbol{C} \rangle$, where $\boldsymbol{A}$ and $\boldsymbol{C}$ are the antecedent and consequent of $\boldsymbol{\Upsilon}$, and they contain feature assignments in the form of conjunctive formulas.

A product $\mathbf{P}$ *satisfies* a feature formula (such as $\boldsymbol{A}$ or $\boldsymbol{C}$) only if all the feature assignments in the formula are defined in $\mathbf{P}$. The support of an association rule $\boldsymbol{\Upsilon}$ is defined as the number of products that satisfy $\boldsymbol{A} \wedge \boldsymbol{C}$ divided by the number of all products, i.e.,

$$\text{Supp}(\boldsymbol{\Upsilon}: \boldsymbol{A} \rightarrow \boldsymbol{C}) = |\{\mathbf{P} \mid \mathbf{P} \textit{ satisfies } \boldsymbol{A} \wedge \boldsymbol{C}\}| \,/\, |\mathbb{P}|$$

According to [2], an association rule $\boldsymbol{\Upsilon}$ is constructed from a frequent itemset $\boldsymbol{\Gamma}$ by splitting its feature assignments into $\boldsymbol{A}$ and $\boldsymbol{C}$ arbitrarily. Therefore, the support of $\boldsymbol{\Gamma}$ equals to the support of $\boldsymbol{\Upsilon}$. For instance, the association rule $\langle$**F**1="defined"$\rightarrow$**F**3=512$\rangle$ is derived from the frequent itemset {**F**1=defined, **F**3=512} with the same support 2/3. Since the support of $\boldsymbol{\Gamma}$ satisfies its threshold *minsup*, the support of $\boldsymbol{\Upsilon}$ also satisfies *minsup*, and hence *minsup* of $\boldsymbol{\Gamma}$ is also considered as *minsup* of $\boldsymbol{\Upsilon}$. Moreover, a strong association rule should further satisfy a minimum confidence (*minconf*). The confidence of an association rule $\boldsymbol{\Upsilon}$ is defined as the number of products that satisfy $\boldsymbol{A} \wedge \boldsymbol{C}$ divided by the number of products that only satisfy $\boldsymbol{A}$, i.e.,

$$\text{Conf}(\boldsymbol{\Upsilon}: \boldsymbol{A} \rightarrow \boldsymbol{C}) = |\{\mathbf{P} \mid \mathbf{P} \textit{ satisfies } \boldsymbol{A} \wedge \boldsymbol{C}\}| \,/\, |\{\mathbf{P} \mid \mathbf{P} \textit{ satisfies } \boldsymbol{A}\}|$$

For instance, the confidence of the association rule $\langle$**F**1="defined"$\rightarrow$**F**3=512$\rangle$ is 2/3=0.67. In fact, the confidence of $\langle \boldsymbol{\Upsilon}: \boldsymbol{A} \rightarrow \boldsymbol{C} \rangle$ has the same definition as the conditional probability of $\boldsymbol{C}$ given $\boldsymbol{A}$, i.e., Prob($\boldsymbol{C}|\boldsymbol{A}$). Besides, the value of *minconf* is usually larger than 0.5 and given by domain experts.

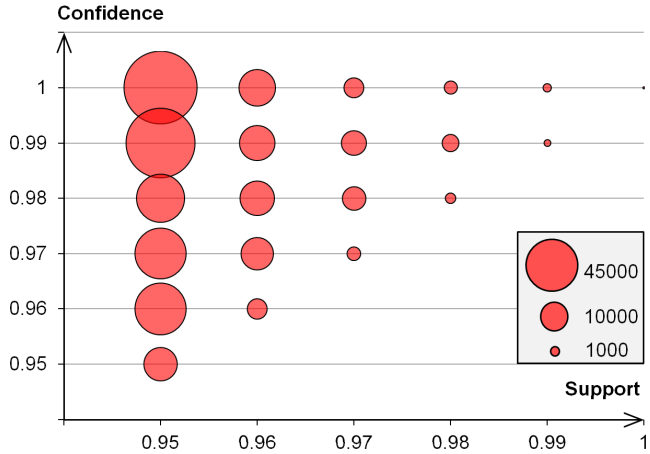


Figure 6. Support and Confidence of Association Rules.

In our industrial example, the *minsup* remains to be 0.95 and the *minconf* is set to be also 0.95. Finally, we find 228,830 association rules based on the previous calculated 4029 frequent itemsets. These rules involve 10 binary features and 8 non-binary features. We group these rules by their support and confidence, and the distribution result is illustrated with a bubble chart in Fig. 6. It shows that most rules have very high confidence, and the largest five groups of rules are presented in Table 2(a), where the confidence of the largest group (containing 45682 rules) even reaches 1. It means that these rules apply to all existing products.

Table 2. Largest Five Groups of Association Rules

Supp	Conf	# Rules
0.95	1.00	45682
0.95	0.99	41715
0.95	0.96	23004
0.95	0.97	22356
0.95	0.98	20169

(a) Grouped by Supp and Conf

Ante.	Cons.	# Rules
3	3	18100
3	4	17360
4	3	17360
2	4	13575
4	2	13575

(b) Grouped by Ante. and Cons.

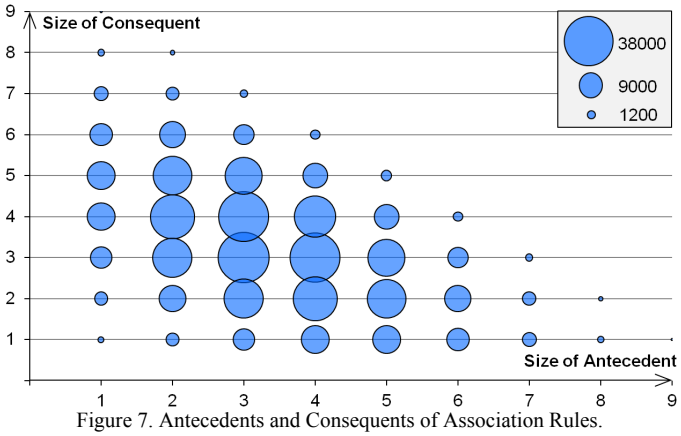


Figure 7. Antecedents and Consequents of Association Rules.

Moreover, we also group all association rules by their size of antecedents (Ante.) and size of consequents (Cons.), and the distribution result is illustrated with a bubble chart in Fig. 7. According to the list of the largest five groups shown in Table 2(b), the biggest bubble indicates a group of 18100 rules involving six features in total. In general, it shows that most rules have around three features both in their antecedents and in consequents. These rules indicate complex correlations between feature assignments, and can provide prediction knowledge to facilitate product configuration in practice.

Besides, it seems that two association rules with symmetrical antecedent and consequent (i.e., $\boldsymbol{A}\rightarrow\boldsymbol{C}$ and $\boldsymbol{C}\rightarrow\boldsymbol{A}$) usually satisfy both *minsup* and *minconf*. The reason is that on the one hand they have the same support that is very high (between 0.95 and 1); on the other hand their confidences depend on Supp($\boldsymbol{A}$) and Supp($\boldsymbol{C}$), which are even higher and close (between Supp($\boldsymbol{A}\wedge\boldsymbol{C}$) and 1). As a result, the distribution of association rules in Fig. 7 looks symmetrical as well.

VI. CORRELATION PRUNING

The association rules satisfying the constraints of *minsup* and *minconf* are considered as potential feature correlations. However, the number of derived association rules can easily explode in a dense dataset. It results into a huge number of redundant association rules. For example, it is unrealistic to understand and validate the 228,830 association rules in our industrial example. Given such a problem, we propose to prune sub-rules that have equal or smaller confidence than their parent rules.

Let $\boldsymbol{Y}$p: $\boldsymbol{A}_\mathrm{p}\rightarrow\boldsymbol{C}_\mathrm{p}$ and $\boldsymbol{Y}$s: $\boldsymbol{A}_\mathrm{s}\rightarrow\boldsymbol{C}_\mathrm{s}$ be two association rules, where $\boldsymbol{A}$ and $\boldsymbol{C}$ denote the antecedent formula and the consequent formula of an association rule. Let $\boldsymbol{S}(\boldsymbol{A})$ and $\boldsymbol{S}(\boldsymbol{C})$ be the itemsets of $\boldsymbol{A}$ and $\boldsymbol{C}$. Assuming $\boldsymbol{S}(\boldsymbol{A}_\mathrm{p})$ is a subset of $\boldsymbol{S}(\mathrm{A}_\mathrm{s})$, and $\boldsymbol{S}(\boldsymbol{C}_\mathrm{s})$ is a subset of $\boldsymbol{S}(\boldsymbol{C}_\mathrm{p})$, we have $\boldsymbol{A}_\mathrm{s} = \boldsymbol{A}_\mathrm{p} \wedge \boldsymbol{A}'$ and $\boldsymbol{C}_\mathrm{p} = \boldsymbol{C}_\mathrm{s} \wedge \boldsymbol{C}'$, where $\boldsymbol{A}'$ and $\boldsymbol{C}'$ are conjunctions of itemsets that can be empty. According to the theorems in propositional logic,

$$\boldsymbol{A}_\mathrm{p}\rightarrow\boldsymbol{C}_\mathrm{p} = \neg\boldsymbol{A}_\mathrm{p}\vee\boldsymbol{C}_\mathrm{p} = \neg\boldsymbol{A}_\mathrm{p}\vee(\boldsymbol{C}_\mathrm{s}\wedge\boldsymbol{C}') = (\neg\boldsymbol{A}_\mathrm{p}\vee\boldsymbol{C}_\mathrm{s})\wedge(\neg\boldsymbol{A}_\mathrm{p}\vee\boldsymbol{C}') \Rightarrow \neg\boldsymbol{A}_\mathrm{p}\vee\boldsymbol{C}_\mathrm{s} \Rightarrow \neg\boldsymbol{A}_\mathrm{p}\vee\neg\boldsymbol{A}'\vee\boldsymbol{C}_\mathrm{s}$$

and

$$\boldsymbol{A}_\mathrm{s}\rightarrow\boldsymbol{C}_\mathrm{s} = \neg\boldsymbol{A}_\mathrm{s}\vee\boldsymbol{C}_\mathrm{s} = \neg(\boldsymbol{A}_\mathrm{p}\wedge\boldsymbol{A}')\vee\boldsymbol{C}_\mathrm{s} = (\neg\boldsymbol{A}_\mathrm{p}\vee\neg\boldsymbol{A}')\vee\boldsymbol{C}_\mathrm{s} = \neg\boldsymbol{A}_\mathrm{p}\vee\neg\boldsymbol{A}'\vee\boldsymbol{C}_\mathrm{s}$$

So, $\boldsymbol{A}_\mathrm{p}\rightarrow\boldsymbol{C}_\mathrm{p} \Rightarrow \boldsymbol{A}_\mathrm{s}\rightarrow\boldsymbol{C}_\mathrm{s}$

Finally, it is concluded that for two association rules $\boldsymbol{Y}$p and $\boldsymbol{Y}$s, if $\boldsymbol{S}(\boldsymbol{A}_\mathrm{p})\subseteq\boldsymbol{S}(\boldsymbol{A}_\mathrm{s})$ and $\boldsymbol{S}(\boldsymbol{C}_\mathrm{s})\subseteq\boldsymbol{S}(\boldsymbol{C}_\mathrm{p})$ then $\boldsymbol{Y}$s is a sub-rule of $\boldsymbol{Y}$p, i.e. $\boldsymbol{Y}\mathrm{p}\rightarrow\boldsymbol{Y}$s. If the confidence of a sub-rule $\boldsymbol{Y}$s is equal or smaller than the confidence of its parent rule $\boldsymbol{Y}$p, i.e., Conf($\boldsymbol{Y}$s)$\leqslant$Conf($\boldsymbol{Y}$p), then $\boldsymbol{Y}$s should be pruned because it does not provide any predictive advantage. For instance in the example of Table 1, if the *minsup* is small enough, then there is an association rule $\langle\mathbf{F2}{=}30\rightarrow\mathbf{F3}{=}512\wedge\mathbf{F4}{=}\mathrm{EUR}\rangle$ with a confidence of 1, and its sub-rule $\langle\mathbf{F2}{=}30\rightarrow\mathbf{F3}{=}512\rangle$ has the same confidence of 1. Therefore, the sub-rule $\langle\mathbf{F2}{=}30\rightarrow\mathbf{F3}{=}512\rangle$ should be pruned from the set of association rules, which does not affect the prediction capacity of feature correlations at all.

After analyzing the previously derived association rules in our industrial example, we finally find 228,377 redundant rules in total and successfully reduce the number of association rules from 228,830 to 453 (0.2%) without losing any predictive knowledge. These rules involve 10 binary features and 8 non-binary features, which has exactly the same coverage as the association mining result before pruning. These binary features and non-binary features are listed in Table 3, and the feature

names are anonymized for the sake of industrial confidentiality. Due to the high threshold of *minsup* and *minconf*, these features are selected in almost all products, and the non-binary features do not have too many different values.

Table 3. Features in Correlations

Feature	# Products
BF1	100
BF2	100
BF3	99
BF4	99
BF5	98
BF6	96
BF7	96
BF8	96
BF9	95
BF10	95

(a) Binary Features

Feature	# Values	# Products
NBF1	1	100
NBF2	1	100
NBF3	1	100
NBF4	2	100
NBF5	1	99
NBF6	1	99
NBF7	2	98
NBF8	1	97

(b) Non-Binary Features

Table 4. Largest Five Groups of Rules After Pruning

Supp	Conf	# Rules
0.95	0.95	80
0.95	0.96	48
0.96	0.96	48
0.97	0.97	36
0.96	0.97	30

(a) Grouped by Supp and Conf

Ante.	Cons.	# Rules
1	7	112
1	6	98
1	8	81
1	5	48
1	9	30

(b) Grouped by Ante. and Cons.

As done in section 5, we again group the remaining association rules first by their support and confidence, and then by their size of antecedent and size of consequent, and the largest five groups in each case are listed in Table 4. The rule distribution in terms of support and confidence is illustrated in a bubble chart in Fig. 8, and the largest bubble is a group of 80 association rules as shown in Table 4(a). Compared to the distribution illustrated in Fig. 6, it indicates that the average association rule after pruning has a similar level of support, but lower level of confidence.

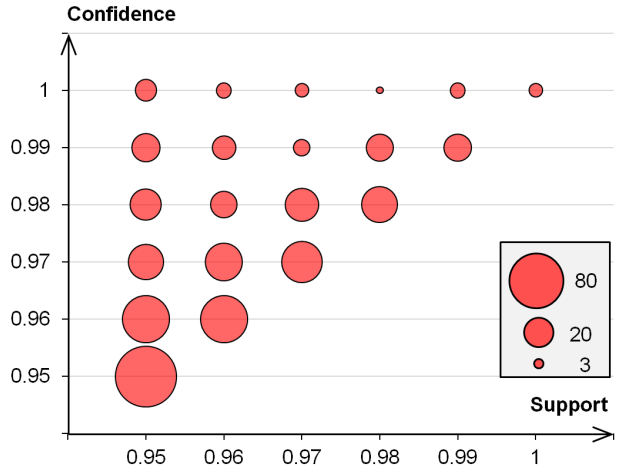


Figure 8. Support and Confidence of Remaining Rules.

The rule distribution in terms of size of antecedent and size of consequent is illustrated in a bubble chart in Fig. 9, and the largest bubble is a group of 112 association rules as shown in Table 4(b). Compared to the distribution illustrated in Fig. 7, it indicates that the average association rule after pruning has a smaller size of antecedent but a significantly larger size of consequent. In fact, that is exactly the characteristics of parent association rules that provide equal or stronger predictive knowledge than their sub-rules.

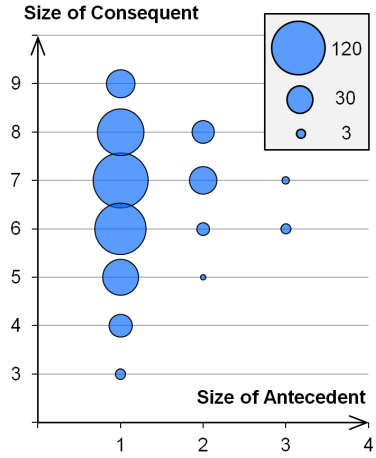


Figure 9. Antecedents and Consequents of Remaining Rules.

VII. RELATED WORK

Since feature correlations are often documented in feature models as feature constraints, there are several related studies in extracting the feature model from existing products of a SPL. Czarnecki et al. [7] [18] introduced the concept of Probabilistic Feature Model (PFM) and extracted soft and hard feature constraints from product configurations. However, they only considered correlations of binary features without value, and the antecedents of their correlations only contain a single feature. Lora-Michiels et al. [13] also proposed a reverse engineering approach to extracting a feature model including structural and transversal feature correlations from product configurations. However, the correlations they identified are only between two binary features. Both of the two studies used association mining techniques to identify feature correlations. However, neither of them considered complex correlations between sets of multiple feature assignments.

Besides, She et al. [19] presented an approach to extracting feature models from feature correlations and descriptions. They proposed to determine parent features based on text similarity and domain knowledge. Acher et al. [1] extract feature models of single products from product descriptions and automatically merge them into a feature model of the SPL. These studies did not address the problem of feature correlation mining from configurations.

Regarding association mining techniques, the classical algorithm is Apriori [2]. However, it is not the optimal solution and has a scalability problem in our previous study [23]. An advanced algorithm for calculating frequent itemsets is LCM [21], which manages to calculate frequent closed itemsets in polynomial time per itemset. Due to the downward closure property, the frequent itemsets can be calculated by finding all subsets of a frequent closed itemset. Our analysis has proved

that this algorithm is capable to handle large dataset, at lease when the minimum support is high.

Regarding correlation pruning, Bayardo et al. [3] proposed to prune the association rules that do not offer a significant predictive advantage. They introduced an additional constraint called improvement, which is defined as the minimum difference between the confidence of a rule and the confidence of any sub-rule. They argued that every association rule shall contribute to its predictive ability with positive improvement, and a setting of the minimum improvement is required as a constraint threshold to further prevent rules with marginal predictive advantage. However, the parent rule and the sub-rule in their definition must have the same consequent, and they only checked if the parent rule and the sub-rule have a subset relationship in their antecedents. In fact, a rule and its sub-rules can have a subset relationship both in their antecedents and consequents. In order to remedy the rule explosion problem, it is important to compare all combinations of rules and their sub-rules and prune all sub-rules with zero or negative improvement.

VIII. CONCLUSION

In this paper, we identified feature correlations from existing product configurations in order to guide product configuration processes. A solution framework is presented with a series of processes, and association mining techniques are used to extract strong association rules as potential feature correlations. Then these correlations are pruned by removing the association rules that do not provide any predictive advantage. At last the remaining correlations are validated by domain experts. All the above processes except correlation validation are conducted automatically.

This approach is demonstrated on a large-scale industrial SPL in the embedded system domain, and finally 453 feature correlations with high support and confidence are derived, and most correlations have multiple feature assignments in their antecedents and consequents. Considering the missing or inconsistent feature correlations in SPL specifications, these feature correlations can be used to provide prediction knowledge and to improve the correctness and efficiency of product configuration processes.

ACKNOWLEDGMENT

This work is within the MOTION project of “Innovationszentrum Applied System Modeling”, sponsored by the German state of Rhineland-Palatinate and Fraunhofer IESE. See http://www.applied-system-modeling.de/.

REFERENCES

[1] M. Acher, A. Cleve, G. Perrouin, P. Heymans, C. Vanbeneden, P. Collet, and P. Lahire, "On extracting feature models from product descriptions," in Proceedings of the Sixth International Workshop on Variability Modeling of Software-Intensive Systems, ser. VaMoS '12. New York, NY, USA: ACM, 2012, pp. 45-54.

[2] R. Agrawal and R. Srikant, "Fast algorithms for mining association rules in large databases," in Proceedings of the 20th International Conference on Very Large Data Bases, ser. VLDB '94. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1994.

[3] R. J. Bayardo, R. Agrawal, and D. Gunopulos, "Constraint-Based rule mining in large, dense databases," in Proceedings of the 15th International Conference on Data Engineering, ser. ICDE '99. Washington, DC, USA: IEEE Computer Society, 1999.

[4] A. Ceglar and J. F. Roddick, "Association mining," ACM Comput. Surv., vol. 38, no. 2, Jul. 2006.

[5] K. Czarnecki, S. Helsen, and U. W. Eisenecker, "Formalizing cardinality-based feature models and their specialization," Software Process: Improvement and Practice, vol. 10, no. 1, pp. 7-29, 2005.

[6] K. Czarnecki and A. Wasowski, "Feature diagrams and logics: There and back again," in Proceedings of the 11th International Software Product Line Conference, ser. SPLC '07. Washington, DC, USA: IEEE Computer Society, 2007, pp. 23-34.

[7] K. Czarnecki, S. She, and A. Wasowski, "Sample spaces and feature models: There and back again," in Proceedings of the 2008 12th International Software Product Line Conference, ser. SPLC '08. Washington, DC, USA: IEEE Computer Society, 2008, pp. 22-31.

[8] S. Deelstra and M. Sinnema, "Managing the complexity of variability in software product families," Ph.D. dissertation, Institute of Mathematics and Computing Science, University of Groningen, Jul. 2008.

[9] G. Dong and J. Li, "Interestingness of discovered association rules in terms of Neighborhood-Based unexpectedness," in Proceedings of the Second Pacific-Asia Conference on Research and Development in Knowledge Discovery and Data Mining, ser. PAKDD '98. London, UK, UK: Springer-Verlag, 1998, pp. 72-86.

[10] K. C. Kang, S. G. Cohen, J. A. Hess, W. E. Nowak, and A. S. Peterson, "Feature-Oriented domain analysis (FODA) feasibility study," Software Engineering Institute, Carnegie Mellon University Pittsburgh, PA., Tech. Rep. CMU/SEI-90-TR-21, Nov. 1990.

[11] C.W. Krueger, “New Methods behind a New Generation of Software Product Line Successes”, In K.C. Kang, V. Sugumaran, and S. Park, “Applied Software Product Line Engineering”, Auerbach Publications, 2010, pp. 39-60.

[12] The LCM tool. http://research.nii.ac.jp/~uno/codes.htm. (Jan 2013)

[13] A. Lora-Michiels, C. Salinesi, and R. Mazo, "A method based on association rules to construct product line models," in VaMoS'10, 2010, pp. 147-150.

[14] J. D. McGregor, “The evolution of product line assets,” Tech. Rep., 2003.

[15] The Orange Tool. http://orange.biolab.si/. (Oct 2012)

[16] T. Patzke, M. Becker, M. Steffens, K. Sierszecki, J. E. Savolainen, and T. Fogdal, "Identifying improvement potential in evolving product line infrastructures: 3 case studies," in Proceedings of the 16th International Software Product Line Conference - Volume 1, ser. SPLC '12. New York, NY, USA: ACM, 2012, pp. 239-248.

[17] K. Pohl, G. Böckle, and F. J. Linden, Software Product Line Engineering: Foundations, Principles and Techniques. Secaucus, NJ, USA: Springer-Verlag New York, Inc., 2005.

[18] S. She, "Feature model mining," Master's thesis, 2008.

[19] S. She, R. Lotufo, T. Berger, A. Wkasowski, and K. Czarnecki, "Reverse engineering feature models," in Proceedings of the 33rd International Conference on Software Engineering, ser. ICSE '11. New York, NY, USA: ACM, 2011, pp. 461-470.

[20] M. Svahnberg and J. Bosch, "Evolution in software product lines: Two cases," Journal of Software Maintenance, vol. 11, no. 6, pp. 391-422, Nov. 1999.

[21] T. Uno, M. Kiyomi, and H. Arimura, "LCM ver. 2: Efficient mining algorithms for Frequent/Closed/maximal itemsets," in Proceedings of 2nd Workshop on Frequent Itemset Mining Implementations (FIMI'04), ser. CEUR Workshop Proceedings, vol. 126. CEUR-WS.org, 2004.

[22] B. Zhang and M. Becker, "Code-based variability model extraction for software product line improvement," in Proceedings of the 16th International Software Product Line Conference - Volume 2, ser. SPLC '12. New York, NY, USA: ACM, 2012, pp. 91-9.

[23] B. Zhang and M. Becker, "Mining complex feature correlations from software product line configurations," in Proceedings of the Seventh International Workshop on Variability Modelling of Software-intensive Systems, ser. VaMoS '13. New York, NY, USA: ACM, 2013.