Customer buying behaviour analysis in mass customization

Tilak Raj Singh [1] and **Narayan Rangaraj** [2]

Abstract. Motivated by the importance of customer buying behaviour (such as correlation among product attributes/features of products configured in the past) in planning future configurations, this paper addresses the issue that product evolution (upgrades) usually render information gathered from past buying behaviour at least partially unusable. For instance, relations among features might have been changed, thus making it difficult to configure the same products again. The proposed approach aims to (1) find associations between product attributes based on the analysis of prior customer orders (2) apply configuration rules to prune attribute association rules which are not controlled by customers, and (3) check whether derived attribute association rules from past orders also work for the new upgraded product. Attribute associations consistent with the upgraded product are then used to predict configurations for production planning. We use machine learning algorithms and optimization techniques to address these issues.

1 Introduction

Mass customized products (e.g. Automobiles) involve a large number of product variants which are generated by combining different predefined features/attributes. Individual product attributes and attribute combinations control the final consumption of vehicle components and sub-assemblies during the production [13]. For example, the selection of features such as a specific gearbox or a sports package can decide which steering wheel will be used to build the vehicle. Thus, knowledge of customer buying behaviour (correlation between product attributes) is crucial for demand estimation of parts and sub-assemblies for future production.

One way to get customer buying behaviour is by extrapolating the configurations produced in the past. Due to the high degree of individualization and continuous changes in the product design, product evolution (upgrades) usually renders information gathered from past buying behaviour at least partially unusable. For instance, relations among features might have been changed, thereby not allowing configuration of the same products again. In the special case of a new product, information from existing models (having common features) is often used to prepare the initial production plan (set of vehicle configurations). As configuration rules of different products are not same, it is likely that *attribute associations* from the existing model may not be directly applicable to the new product. Thus, we need a mechanism to validate whether the derived attribute association rules from past orders also work for the new upgraded product. These consistent attribute association rules could then be used to predict future configurations for production planning [15].

Throughout this paper, we will use the terms *attribute association rules* or *attribute associations* to specify quantities reflecting the joint selection of attributes or conditional section of attributes. For example, figure 1 shows that 67% of configurations contain both attribute 1 and attribute 2.

In practice, the use of specific components in the final product assembly depends on 1) the way they are designed and 2) the way customers select them. Design or engineering related dependencies (or restrictions) are well documented in product's Bill-of-Material (BOM) or configuration rules. As the product development process starts well before the actual production, it is possible to know the product description for a future time (2-3 years in advance) from BOM [16]. However, attribute associations from the customer point of view are not known directly and can only be seen once customers have placed orders. Most manufacturers utilize information from product variants produced in the past to get estimates of customer demands. Customer behaviour can be expressed through associations between different attribute choices (e.g. joint selection) customers have made in the past.

In order to predict configurations for production planning the information from 1) configuration restrictions and 2) customer buying behaviour should be used together. Any conflicting information between these two sources points to an inconsistency in the planning information. Delay in the detection of such discrepancies may result in a wrong mix of parts being produced, thus hampering production efficiency.

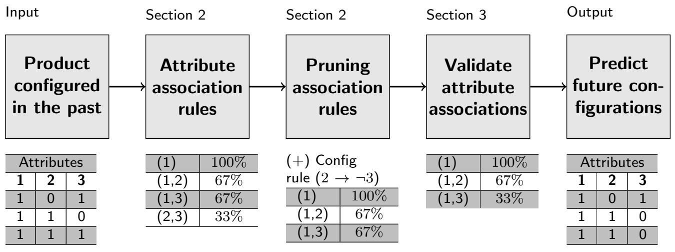


Figure 1. Predicting future configurations as per customer buying behaviour

Figure 1 shows a sequential block diagram for predicting future product configurations as per customer buying behaviour and configuration rules. First, customer prior demand is analysed to calculate attribute association rules (joint and conditional correlation between

[1] IT-Data Analytics, Mercedes-Benz R & D India, Bangalore, Email: tilak.singh@daimler.com
[2] Industrial Engineering and Operations Research, Indian Institute of Technology, Bombay, Powai Mumbai, India, email: narayan.rangaraj@iitb.ac.in

attributes). Then, association rules which conflict with configuration restrictions are pruned. In Section 2 we discuss a machine learning algorithm to calculate such association rules. In Section 3 we present optimization models which aim to build a set of future configurations by considering 1) configuration restriction and 2) attribute association rules simultaneously. If we are able to find such a configuration set which matches both of the input parameters, then the result can be used for future production planning. In case of conflicts between configuration restrictions and attribute association rules, further analysis is required and perhaps only a limited set of consistent association rules can be used to predict future configurations. In this case, our focus is to find such a consistent set of attribute association rules and use them to predict future configurations. Section 4 focuses on the system implementation followed by initial computational results, discussed in section 5.

2 Mining attribute association rules

A customizable product can be configured using different combinations of attributes (features). In an automobile, attribute could be body style, transmission type, sunroof or parking assistance. Customer buying behaviour can be studied by analysing how product attributes are associated with each other. Association rule mining has wide application in data mining for analysing and predicting customer behaviour [13]. In this section, we discuss a framework for mining association rules among product attributes from a given set of product configurations. As association rules are extracted from known product configurations, we first take a look at the characteristics of the configuration problem.

2.1 Product configurations

Let us define our product configuration problem as per [10, Definition 1]: the configuration problem C can be expressed through a triple (X, D, F), where:

- X is a set of product attributes (configuration variables) lets say $\{x_1, ...x_n\}$. Where n is the total number of attributes.
- D is the set of attributes finite domains $d_1, d_2, ..., d_n$.
- $F = \{f_1, f_2, ..., f_m\}$ is a set of propositional formulas (rules or restrictions) over attribute set X.

In this paper, we assume that the configuration variables $x_i \in X$ are boolean, hence domain $d_i \in \{0, 1\}, \forall i \in X$. A configuration is said to be feasible if an assignment for all attributes ($i \in X$) is found which fulfils each and every proposition in F. $X = x_1, ...x_n$ is a set and each d_i is a set. In this case, each d_i is a binary set. In other words, $\{d_1, ..., d_n\}$ is a collection of sets, whereas $\{x_1, ..., x_n\}$ are the elements of set X.

Example 1

Let us assume a car is configured using six attributes $X = \{1, 2, ..., 6\} \equiv \{$ Automatic Gearbox, Cruise control, Reverse camera, Sunroof, Keyless Go, Parktronic $\}$, $D \in \{0, 1\} \forall X$, and $F = \{f_1\}$ where

$f_1 = \{2 \rightarrow 1\}$: Cruise control requires Automatic Gearbox.

For a given set of boolean variables (attributes) and propositional formulas, finding a feasible configuration is a *Boolean Satisfiability* problem where the aim is to get an assignment (true or false value) of Boolean variables (X) which satisfies given configuration rules (F). Configurations from Table 1 can be treated as customer configurations and in the next section we will derive associations between different attributes from these. Table 1 contains a list of some feasible configurations which are created by combining given attributes and satisfying configuration rule F.

Order#	Automatic Gear-Box (AG)	Cruise Control (CC)	Reverse Camera (RC)	Sunroof (SR)	KeyLess Go (KG)	Parktronic (PA)
O001	1	1	1	1	1	0
O002	1	1	0	1	1	0
O003	1	0	1	0	1	0
O004	1	0	0	0	0	1
O005	1	1	1	1	0	1
O006	1	0	0	0	0	1
O007	1	0	1	1	1	1
O008	1	0	0	0	1	1
O009	0	0	1	0	1	1
O010	0	0	1	1	0	0

Table 1. Sample configurations (selected by customers) from Example 1

2.2 Attribute association rules and configuration restrictions

Association rule mining methodology is used to find the association between variables in large transactions [1]. In our case, each configuration is expressed over a subset of attributes, where attributes are feature/variables used to configure the product. An association between two disjoint sets of attributes p and q can be expressed using two numbers:

$Support(p \Rightarrow q)$: This is the proportion of configurations that contain both attribute sets p and q. In *Example1*, Support $(Sunroof,\ ReverseCamera)$= 4/10 = 0.4.
Confidence $(p \Rightarrow q)$: Given set of configuration which contains attributes set p, this is the proportion of configurations where attribute set q is also selected. In *Example1*, Confidence $(Sunroof \Rightarrow ReverseCamera)$ = 4/5 = 0.8. If support and confidence are greater than user specified thresholds, then we call that association rule “interesting”.

After applying rule mining techniques, we get a large number of relations which satisfy our parameter of interestingness, although not all of them are customer driven. Because sales transactions do not explicitly state only customer selectable attributes, it is then our task to identify and remove attribute relations which are driven by the technical nature of the product. For example, in Table 2 (rule # 2) the association between two attributes $CruiseCtrl \Rightarrow AutomaticGearBox$ is given by Support = 0.3 and confidence = 1. The high confidence between Cruise control and Automatic Gearbox is not really driven by customer buying behaviour, but this is the only way a feasible configuration using attribute Cruise control can be created. This information is stated in the configuration rule ($F = \{f_1\}$) of Example 1.

In practical scenarios with hundreds of product attributes and thousands of configuration rules, identifying which attribute association rule is controlled by their technical dependencies is non-trivial. Also,

sr.	lhs		rhs	support	confidence
1	{CC}	⇒	{SR}	0.3	1
2	{CC}	⇒	{AG}	0.3	1
3	{PA}	⇒	{AG}	0.5	0.83
4	{KG}	⇒	{AG}	0.5	0.83
5	{SR}	⇒	{RC}	0.4	0.8
6	{SR}	⇒	{AG}	0.4	0.8
7	{CC}	⇒	{RC}	0.2	0.66
8	{CC}	⇒	{KG}	0.2	0.66
9	{RC}	⇒	{KG}	0.4	0.66
10	{RC}	⇒	{AG}	0.4	0.66
11	{SR}	⇒	{KG}	0.3	0.6
12	{PA}	⇒	{RC}	0.3	0.5
13	{PA}	⇒	{KG}	0.3	0.5
14	{SR}	⇒	{PA}	0.2	0.4
15	{CC}	⇒	{PA}	0.1	0.33

Table 2. Association rules between two attributes from Example 1

it is both error prone and time consuming to analyse and classify a large set of attribute associations manually. As the configuration problem is used to find feasible assignments of product attributes under configuration rules, we use this to state some direct dependencies among attributes. Any two attributes (let's say p and q) can be combined in a configuration based on the following relations:

$Sr\#$	Relation	p	q	Description
1	$\neg p \wedge \neg q$	0	0	Configuration without attribute p and q
2	$\neg p \wedge q$	0	1	Configuration with attribute q but not p
3	$p \wedge \neg q$	1	0	Configuration with attribute p but not q
4	$p \wedge q$	1	1	Configuration with both attribute p and q

Table 3. Possible relationships among two attributes in a configuration

Depending upon how many relations are satisfied from Table 3 any association rule can be classified in one of 2^4 possible cases. For example, if we consider Cruise control and Automatic gearbox from example 1, as attribute p and q, then with the given configuration rule, we will not be able to create a configuration which satisfies 3^{rd} relation $p \wedge \neg q$. If all the four relations are satisfied from Table 3 then we can say that the given attributes are independent of product's technical influence and any association derived from customer orders actually reflects their buying behaviour.

In the other case, let us assume that the product from Example 1 has been upgraded and new configuration restriction has been added i.e. $F = \{f_1, f_2\}$ where f_2= Parktronic comes with Reverse camera. Due to this new restriction configuration, O001, O003 and O010 from Table 1 will not be feasible for future product. Then, associations between different product attributes need to be validated against the new configuration rule. In the next section, we discuss a set of optimization models for validating attribute association rules with respect to configuration restrictions.

3 Validation of attribute association rules

In the task of validating attribute association rules for an upgraded product; our aim is to find one instance of future demand where all derived association rules are satisfied. The future demand estimate can be given in terms of a configuration set where correlation between attributes is controlled by predefined association rules.

If products are defined over sets of boolean attributes, the association rules can be expressed as boolean proposition formulas. For example, $support(p, q)$ can be modelled in a proposition formula to capture the joint selection of the attribute sets p and q. The value of support (p, q) indicates the fraction at which corresponding boolean propositional formula ($p \wedge q$) evaluates as true in the configurations set. Thus, finding a consistent future demand estimate with respect to association rules is equivalent to satisfying a set of propositional formulas with some probability. For example, if $support(p, q) = 0.2$ then we want a clause ($p \wedge q$) is true 20% of the time in final configurations. The resultant set of propositional formulas can be divided into two sets: 1) propositional formulas derived from the configuration problem (i.e. BOM) which has to evaluate to "true" for every demand instance 2) propositional formulas generated from association rules which are assigned a probability of being satisfied. If we are able to find a probability distribution on the truth assignments of the boolean variables corresponds to association rules that induces the given probabilities, then we will have an instance of future demand where all calculated association rules are satisfied.

For a given set of boolean variables and set of boolean clauses, determining whether it is possible to find a probability measure over truth assignments of the boolean variables that induce the given assessments is known as Probabilistic Satisfiability (PSAT) problem [9]. After modelling association rules as boolean propositional formulas and generating a solution instance where all association rules and configuration restrictions are satisfied simultaneously, we can say that derived association rules are consistent with the new configuration restrictions for the product. Our aim is to construct a configuration set such that it reflects the same support and confidence values from association rules as derived from past data. Support and confidence can be modelled as constraints in configuration problem and the associated quantity is then used to select the configuration set/ solution set to check the satisfiability [17].

3.1 The association rules verification model

In this section, we present the optimization model for evaluating the consistency among attributes association rules. Before formulating the mathematical model, let us discuss a small example to understand the underlying problem:

Example 2: *Let us assume we have discovered a customer behaviour from Table 1 that three different attributes (Reverse Camera (RC), Keyless Go (KG), Parktronic (PA)) are individually selected* 60% *of the time in prior demand. Now, new configuration restrictions (e.g. Upgraded product) specify that at least two of the attributes (out of three) have to be present in every feasible configuration. Now, is it feasible to assume that the attributes will be selected at the same rate as before?*

From given configuration rule in Example 2, at least two attributes have to be present in a feasible configuration, i.e. the following boolean clause has to be true: (RC $\vee$ KG), (KG $\vee$ PA) and (RC $\vee$ PA). If we are able to get a set of configurations where the above rules are satisfied and each attribute is selected 60% of the time in the configuration set, then we can say that derived attribute association rules and configuration restrictions are consistent with the upgraded product.

Before solving the problem of Example 2 let us formulate a general mathematical model to detect consistency among the association rules. The problem of validating association rules with respect to a new configuration rule can be defined as follows:

Let the index i refer to a logical association rule statement (Support or Confidence) defined over n Boolean variables $x_1, x_2, ...x_n$ using Boolean propositional formulas. As an Example from Table 2, x_1= {CC}, x_2= {SR}, the new variable π_1= Support(x_1,x_2) = $x_1 \wedge x_2$.

Let the index j refer to a configuration in the set J, the total configuration set (typically of very large size).

Data

π_i is the probability of i^{th} statement (attributes Support or Confidence) to be true. As an Example from Table 2, π_3=0.4

$$A_{i,j} = \begin{cases} 1 & \text{if } i^{th} \text{ statement is true in } j^{th} \text{ configuration} \\ 0 & \text{otherwise} \end{cases}$$

Decision variables

X_j = Fraction representing the proportion of j in the total configuration set, a real number between 0 and 1;
Z_i^+ = Positive deviation from target probability π_i, a real number between 0 and 1
Z_i^- = Negative deviation from target probability π_i, a real number between 0 and 1

Objective Function

$$OPT_1\textbf{: Minimize} \sum_i Z_i^+ + Z_i^- \tag{1}$$

Subject to

$$\sum_j A_{ij} X_j + Z_i^+ - Z_i^- = \pi_i \ldots \forall i \tag{2}$$

$$\sum_j X_j = 1 \tag{3}$$

$$0 \leq X_j, Z_i^+, Z_i^- \leq 1 \tag{4}$$

3.1.1 *Support*

Assigning support and confidence values to π_i in model OPT_1 is to control the joint probability and conditional probability among attributes. Support of any two attributes association (p $\wedge$ q) can be considered in the model OPT_1 by constraint 2 as follows:

$$\sum_j A_{p\wedge q,j} X_j + Z_{p\wedge q}^+ - Z_{p\wedge q}^- = \pi_{p\wedge q} \tag{5}$$

where $\pi_{p\wedge q}$ is equal to support (p $\Rightarrow$ q). $A_{p\wedge q,j}$ will take value one if both p and q are present in configuration j, otherwise zero.

3.1.2 *Confidence*

Confidence value from association rule mining can be controlled through conditional probability of given attributes. Confidence (p $\Rightarrow$ q) = $support(p \Rightarrow q)/support(q)$ = $\pi_{p|q}$.

$$\sum_j (A_{p\wedge q,j} - \pi_{p|q} A_{q,j}) X_j + Z_{p|q}^+ - Z_{p|q}^- = 0 \tag{6}$$

Above equation controls the confidence (p $\Rightarrow$ q) by controlling the joint selection of p and q and individual selection of attribute q in the configuration.

Decision variables Z_i^+, Z_i^- are used to control the absolute deviation for each association rule. The matrix A lists the set of all feasible configurations which we use to match association rule quantity π_i. At this point let us say that A contains all feasible orders. From model OPT_1 if Z_i^+, Z_i^- are zero $\forall i$ then we can say the association rules are consistent among each other and also with configuration rules as we are only considering feasible configurations in the matrix A.

In Example 2, let us assume that x_1 = RC, x_2 = KG, x_3 = PA. Then the configuration constraint can be written as follows:
x_1 + x_2 = $\geq$ 1, x_1 + x_3 = $\geq$ 1 and x_2 + x_3 = $\geq$ 1.
any combination of x_1, x_2, x_3 which satisfies the above constraint will be a possible configuration to use by model OPT_1 i.e. as a column of A-matrix. In this case, only four possible solutions are available so we can solve this example by explicitly enumerating all possible configurations.

$$OPT_{Example2} : \textbf{Minimize} \sum_{i=1}^{3} Z_i^+ + Z_i^- \tag{7}$$

$$\underbrace{\begin{pmatrix} 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 0 & 1 & 1 & 1 \end{pmatrix}}_{A} \underbrace{\begin{pmatrix} X_1 \\ X_2 \\ X_3 \\ X_4 \end{pmatrix}}_{X} + \underbrace{\begin{pmatrix} Z_1^+ \\ Z_2^+ \\ Z_3^+ \end{pmatrix}}_{Z^+} - \underbrace{\begin{pmatrix} Z_1^- \\ Z_2^- \\ Z_3^- \end{pmatrix}}_{Z^-} = \underbrace{\begin{pmatrix} 0.6 \\ 0.6 \\ 0.6 \end{pmatrix}}_{\pi} \tag{8}$$

$$X_1 + X_2 + X_3 + X_4 = 1 \tag{9}$$

$$0 \leq X_j, Z_i^+, Z_i^- \leq 1 \tag{10}$$

By solving the optimization model $OPT_{Example2}$, at optimality we get objective function value 0.2 ($\neq$ 0). There are multiple optimal solutions and one is $X = 0.4, 0.2, 0.4, 0$ which means configuration 1, 2 and 3 are used 40%, 20% and 40% of the time respectively and configuration 4 is not used in the final solution. As the objective function value of $OPT_{Example2}$ model is not equal to zero, we can say that attribute association and configuration rules are not consistent with each other. However, one drawback of the model OPT_1 is that it does not explicitly specify how many association rules are satisfied. More specifically, we would like to be able to build a model which satisfies the maximum number of association rules in case of conflicting inputs as presented in $Example_2$. In the next subsection, we discuss one such model.

3.2 Maximum association rules fulfilment model

Complex products such as automobile undergo enormous changes through their life cycle. Thus, it is quite likely that some of the association rules derived from past orders may not be consistent with new configurations rules as discussed in our $Example_2$. A relevant question is whether we can maximize the coverage of association rules which can be fulfilled by an upgraded product. The OPT_1 model discussed in section 3.1 can only detect if all the association rules are satisfiable or not. If there are conflicts, we need to ideally find the minimum number of association rules, so that if we remove those, then all the remaining association rules become consistent.

With the same definitions of variables as in the OPT_1 model, let us formulate the problem as follows:

Decision variables:

$$y_i = \begin{cases} 1 & \text{if } i^{th} \text{ association rule statement is not satisfied} \\ 0 & \text{otherwise} \end{cases}$$

Objective Function

$$OPT_2 = \textbf{Minimize} \sum_i y_i \quad (11)$$

Subject to

$$\sum_j A_{ij}X_j + Z_i^+ - Z_i^- = \pi_i \ldots \forall i \quad (12)$$

$$\sum_j X_j = 1 \quad (13)$$

$$y_i \geq Z_i^+ + Z_i^- \ldots \forall i \quad (14)$$

$$0 \leq X_j, Z_i^+, Z_i^- \leq 1; y_i \in \{0,1\} \quad (15)$$

y_i is a binary 0-1 decision variable associated with each association rule (confidence/ support). The variable y_i will take value 1 if there is some deviation between selection of attribute association $\sum_j A_{ij}X_j$ and the given rate π_i. For any association rule at a time only one variable Z_i^+ or Z_i^- will have a non zero value. Accordingly, y_i will take value 0 or 1 from constraint 14. If all association rules are consistent, y_i must be zero for all i.

In $Example_2$ the OPT_2 model will give objective function value 1 i.e. if we ignore one association rule then we can satisfy the remaining ones in the new product configuration. For example if we take only $\pi_2, \pi_3 = 0.6$ then we can build a set of feasible configurations which can satisfy both the association rules.

3.3 Solution procedure

Model formulation (OPT_1) and (OPT_2) are designed to express all possible configurations (X). This runs into the hundreds of millions! There are still two different problems with the optimization model (OPT_1) and (OPT_2):

1. How to consider all possible solutions? Very large number of decision variables.
2. How to build $A_{i,j}$ matrix?
   - Do we have to explicitly write all the columns of A?
   - Can we work with a small set of configurations and add more when needed?

The key for success here is that A_{ij} needs not to be stored or built explicitly. Based on the need, configurations can be added to A_{ij} to minimize the objective function. A solution for such a large scale optimization can be found using column generation [8]. We can start with a possible set of X_j variables. Solve OPT_1 or OPT_2 to decide which of those X_j's are included in the solution, and then try to generate a new configuration so as to improve the objective function value.

As discussed in figure 2, the overall problem is divided into two optimization problems. The first problem is to make a selection over known configurations (stored in $A_{ij} - Matrix$) such that attribute association rules can be matched as per objective function. In the next step, we want to know whether there is any feasible configuration (w.r.t. configuration rules) which improves the master problem objective function value. Let us assume that we have an 'Oracle' (submodel) which will give us such a configuration each time we ask the question. If no such configuration is found, we conclude that there is no feasible configuration which can improve our current objective function value. The aim of the master problem in both model OPT_1 and OPT_2 is to find the optimal solution over known configurations (given columns of the A_{ij}-matrix) and the task of the sub model is to add new columns to the A_{ij}-matrix. We will repeat this procedure until no configuration is found that is worth considering. In the next section, we will discuss the formulation of sub model w.r.t. master problem OPT_1.

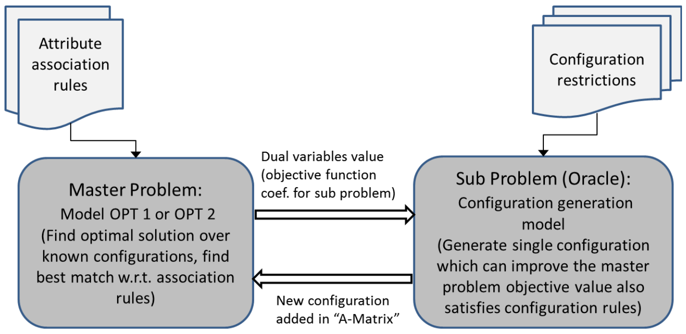

Attribute association rules
Configuration restrictions
Master Problem: Model OPT 1 or OPT 2 (Find optimal solution over known configurations, find best match w.r.t. association rules)
Dual variables value (objective function coef. for sub problem)
New configuration added in "A-Matrix"
Sub Problem (Oracle): Configuration generation model (Generate single configuration which can improve the master problem objective value also satisfies configuration rules)

Figure 2. Column generation procedure to solve OPT_1 and OPT_2

3.4 The configuration generation model (Sub model)

In linear programming problems, in each iteration of the simplex algorithm we compute reduced costs to check if any non-basic variable can enter as part of the solution. To do so in model OPT_1, we have to evaluate if $-\sum w_i * [x_i]_j < 0$ for any configuration j. Where w_i is dual variable and $[x_i]_j$ is the new j^{th} configuration. As the value of w_i will be known from solving model OPT_1, if we know the coefficient of j^{th} order $[x_i]_j$ we can tell whether the given order will improve our objective function or not. Another way to look at this problem is to build the new j^{th} configuration so that $\sum w_i * [x_i]_j$ can be maximized.

The Sub-Problem:
Data: w_i= Dual variable from the model OPT_1, associated with constraints 2
B= Set of constraints derived from configuration rules
Decision Variable

$$[x_i]_j = \begin{cases} 1 & \text{if } i^{th} \text{ attributes association is true in new configuration j} \\ 0 & \text{otherwise} \end{cases}$$

$[x_i]_j$= new configuration for j^{th} column of configuration matrix $A_{i,j}$

Objective:

$$OPT1_{Sub} = \textbf{Maximize} \sum_i w_i * x_i \quad (16)$$

subject to:

$$B[x] \leq b \quad (17)$$

(i.e. x is a feasible configuration)

$$x_i = \{0, 1\} \tag{18}$$

In this formulation, we assume that configuration restrictions are modelled as a set of linear constraints as per Eq. 17 [15]. The subproblem will generate a possible new configuration j. If this new configuration j satisfies Eq. 19 the configuration j enter the pool. This will be one column of A_{ij} matrix in OPT_1 model. Each solution of $OPT1_{Sub}$ model will give a feasible configuration after satisfying all configuration rules from constraint 17. Configuration rules can be presented as propositional formulas and then transformed to sets of linear constraints [15].

$$\sum_i w_i * x_i \geq 0 \tag{19}$$

Dual costs are recomputed by solving the model OPT_1 and the process terminates when no more configurations are found to be worth taking in. We use IBM ILOG Cplex engine to solve the sub-problem. The master (OPT_1) and the subproblem ($OPT1_{Sub}$) may have to be solved multiple times before the terminating criteria is satisfied.

3.5 Column generation

The procedures discussed in section 3.1 and 3.4 works together to find a set of consistent association rules. Model OPT_1 works with a predefined set of configurations and optimizes the current deviation with given association rules target. Model $OPT1_{Sub}$ is used to find a new configuration so that Model OPT_1 objective function value improves. Implementing the column generation approach in association rule verification problem is done in the following way:

1 Solve the OPT_1 model with current columns of A_{ij} matrix. In the first iteration, a feasible configuration can be used to initialize the A matrix. This iteration is used to get the dual variables which will be used in the new configuration generation model $OPT1_{Sub}$.
2 Get dual variable w_i from Eq. 2 of the OPT_1 Model
3 Set up a sub problem as per Section 3.4
4 Get new column (order as 0-1 vector $[x]$ from solution of the sub problem discussed in section 3.4)
5 This generates a possible new configuration j with dual variable w_i
6 If configuration j satisfies $\sum_i w_i * x_i \geq 0$ (pricing inequality) then configuration j enters as j^{th} column of A_{ij}
7 Dual costs are re-computed and the process terminates when no more configurations satisfy the pricing inequality.

4 Implementation

One of our goals is to provide a software system which can 1) extract association rules from given configurations and is 2) able to use new configuration rules (upgraded product) to validate attribute association rules. Figure 3 shows the implementation flow of arriving customer driven attribute associations for predicting future configurations. We use *Apriori* algorithm through R-arules package to find a list of interesting (by support and confidence threshold) association among product attributes [11]. All derived association rules are then used in the optimization model which is implemented using IBM ILOG Cplex 12.5 and $c\#$ .net environment. At the end, a list of all consistent association rules is available with the corresponding set of configurations.

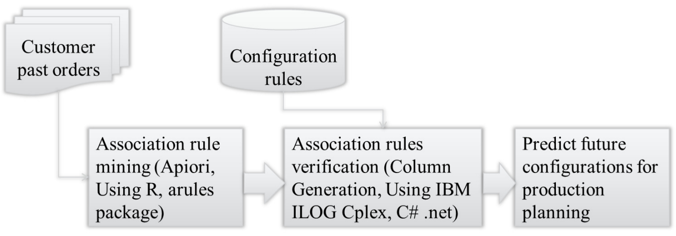
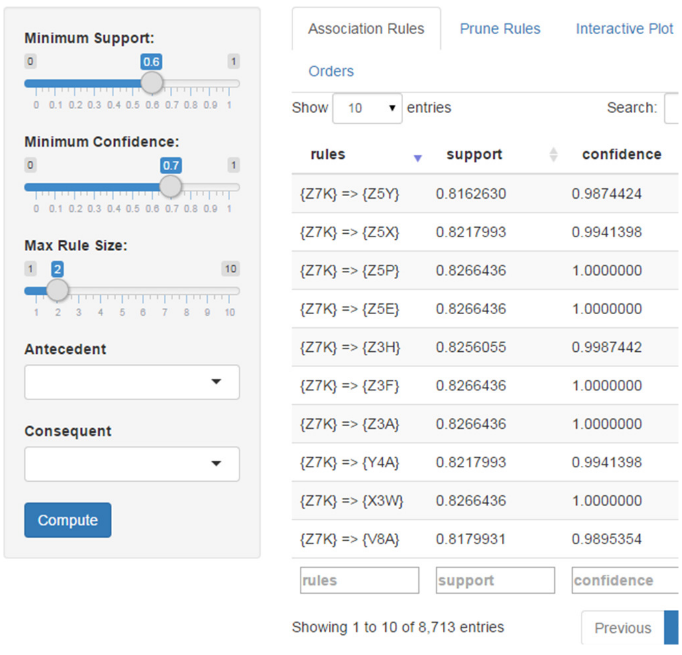

Customer past orders
Configuration rules
Association rule mining (Apiori, Using R, arules package)
Association rules verification (Column Generation, Using IBM ILOG Cplex, C# .net)
Predict future configurations for production planning

Figure 3. Implementation flow of customer drive attributes association rule mining

4.1 Association rule mining

We use Apriori algorithm implemented in "arules" package of R to derive a list of association rules. Number of attributes present in all customer orders are in the range of 500- 1,000. Total number of order is in the range of 10K to 1 Million. Even for very high cut off of support and confidence value (80%) number of generated association rules ranges in tens of thousands. One way to reduce the number



Minimum Support: 0.6

Minimum Confidence: 0.7

Max Rule Size: 2

Antecedent

Consequent

Show 10 entries Search:

rules	support	confidence
{Z7K} => {Z5Y}	0.8162630	0.9874424
{Z7K} => {Z5X}	0.8217993	0.9941398
{Z7K} => {Z5P}	0.8266436	1.0000000
{Z7K} => {Z5E}	0.8266436	1.0000000
{Z7K} => {Z3H}	0.8256055	0.9987442
{Z7K} => {Z3F}	0.8266436	1.0000000
{Z7K} => {Z3A}	0.8266436	1.0000000
{Z7K} => {Y4A}	0.8217993	0.9941398
{Z7K} => {X3W}	0.8266436	1.0000000
{Z7K} => {V8A}	0.8179931	0.9895354
rules	support	confidence

Showing 1 to 10 of 8,713 entries

Figure 4. System overview: Deriving association rules from prior configurations using R-arules

of association rules is by limiting the attribute association level. We usually limit the association between 2 or 3 three attributes. Figure 4 shows a screen shot of actual system for association rule mining. The user can select different computational parameters such as minimum support and confidence, rule size, limiting antecedent (left-hand-side) and consequent attributes to build the desired association rules. Association rule which are according to the user specified parameters are computed from given configuration set. These association rules are used as the target to predict future product configurations.

4.2 Predicting configuration set

As a next step after computing attribute associations (support and confidence), we build the optimization model as per section 3 to compute configurations as per target input characteristics. Two optimization models are used:

1 *Master model* as per section 3.1 (OPT_1) which models selection of configuration such that sum of absolute deviation from target association rules can be minimized.
2 *Sub model* as per section 3.4 which models all configuration rules as linear inequalities.

Both the optimization model receives input from each other in every iteration of column generation procedure described in section 3.5. Optimization models run iteratively until stopping criteria (no improvement to the current solution) is met. At any iteration of column generation procedure, sub model gives a single configuration which is used as new decision variable to the master problem. We have implemented both the optimization model using cplex 12.5. As a result of the optimization procedure, a configuration set is build adhering given association rule targets. In the next section, we will discuss our first computation result with developed models.

5 Computational Results

In this section, we discuss typical computational parameters and associated numbers with input data and decision variables. We have tested our methods and models mainly on automotive data. The historical configurations are analysed at specific granularity (product-line/body style/market/engine type) to reflect current sales planning. Generally, about 500-1000 unique attributes are to be specified in a configuration set. However, not all attributes are available for customers choice as some of them are related to production. In our analysis, we have considered between 100 and 200 attributes which are available to customers. Three vehicle segments are used to test the methodology that has been developed. Table 4 shows various parameters of the data segments that we have selected. The number of orders is the number of prior configurations used to find the attribute associations (support and confidence). For this experiment, the maximum number of attributes in an association rule is limited to two i.e. association among two arbitrary attributes are computed.

Experiment #	# of attributes	# of orders	# association rules	# Pruned association rules (after applying configuration rules)
$Segment_1$	200	30,000	2,000	1,200
$Segment_2$	120	25,000	1,800	1,300
$Segment_3$	100	10,000	1,500	1,100

Table 4. Computational experiments with three different vehicle segments

Starting from a set of configurations and attributes, we use additional parameters such as minimum support and minimum confidence to compute attribute association rules. In this experiment, our aim is to find association rules which are significant (e.g. above minimum support and confidence support). After applying data mining methods, we get a large number of attribute association rules which are then filtered as per their direct dependencies/conflict with configuration rules as explain in section 2.2. By doing so, we see a significant reduction in the number of association rules. These rules are now ready to be validated with respect to the upgraded product.

Now we apply optimization model discussed in section 3.1 to build a set of configurations so that association and configuration rules are met together. In most of the cases, not all computed association rules are applicable to the upgraded product. Therefore, it is important to use only consistent rules to predict the set of future configurations. We applied the optimization model discussed in section 3.1 to build such a configuration set.

An important use of the set of predicted future configurations is to find part demand estimates for future production. In order to see the influence of consistent customer buying behaviour in parts demand, we computed parts frequency associated with order sets, where 1) only consistent attribute associations are used to predict the order set and 2) all attribute associations available after pruning w.r.t. configuration rules are used to predict the order set. The above association rules are also supplemented with a few sales forecasts (at single attribute level) to include estimates of new attributes which are not present in past orders.

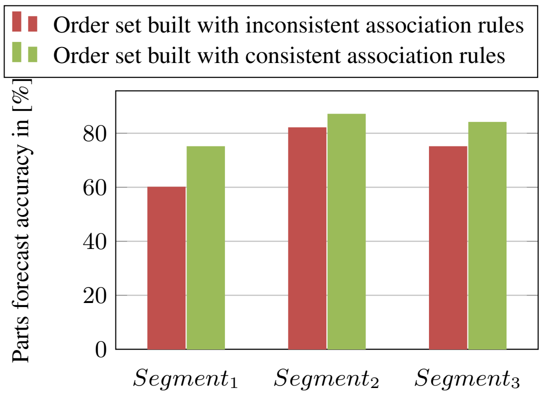


Figure 5. Comparing part demand forecast accuracy between configuration set built with consistent and inconsistent attribute association rules

Figure 5 shows the part forecast accuracy of the two scenarios discussed above. The order sets are compared with real customer orders to find the match with respect to estimated part number. About 10,000 part forecasts are compared and we used part demand matching parameter $\pm 10\%$ i.e. if the estimated value of part demand is within $\pm 10\%$ of actual demand then this forecast is considered good. With this measurement, we have compared two order set computed with consistent and non-consistent attribute association. In figure 5, for all the 3 cases we see significant improvement in forecast accuracy when consistent sets of input information are used.

6 Related work

Data mining techniques in manufacturing system are widely used to provide detailed insights regarding processes and products, such as customer segmentation, production control and quality controls [7]. In mass customization, the uncovering of aggregated level of customer buying information becomes crucial due to high product variety [14]. Data mining techniques such as association rule mining have been used in many applications such as predicting a sub-assembly selection, and lead to significant improvement in order fulfilment process [13]. The main challenge in association rule mining technique is how to find useful association from a large set of possible attribute choices [3]. As association rules are derived from historical demands, another challenge is to validate association rules with respect to engineering changes to the product. Configuration models which capture configuration restrictions can be used to validate the list of association rules. It turns out that validating product attribute association rules against configuration rules can be formulated as Probabilistic Satisfiability Problem (PSAT) [2]. Optimization techniques such as column generation can be used to find a solution of PSAT problem [9].

Another way of building reasoning between product attributes from known configurations is through feature models [4]. The basic idea is to deduce rules/ constraints from existing product variants to support reverse engineering [12]. Feature models, combined with configuration rules, can represent hierarchical relations among different product attributes, modelling a complete set of configurations implicitly [6]. However, our aim is to build a small set of explicit configurations which can be used for production planning. The product comparison matrix is another intuitive way to highlight the differences between two products [5]. However, building such a matrix for different customer buying behaviour is a challenging task.

In our work, we have formulated configuration problem as an optimization model to give an integrated solution within the column generation framework of validating set of association rules. Our model takes support and confidence value to attribute associations to build a set of configurations adhering given input targets. In case of conflicting association rules, the model that has been developed attempts to find the maximum number of association rules which can be satisfied after considering product configuration changes.

7 Future work

In this paper, we have discussed a framework for learning customer buying behaviour through data mining and optimization-based techniques. In mass customization, due to frequent changes in products, we are required to validate product attribute associations learnt from customer prior demand. The association rule mining technique when combined with the configuration problem gives the required framework for calculating consistent and feasible attribute associations. These associations among attributes can be used as inputs for predicting configurations for future production planning. The proposed framework uses data mining libraries from R to find association rules. The integrated framework with optimization models provides the ability to perform tests on many scenarios before using any association discovered from past data to future product planning.

One application of discovering attribute associations is to use them for predicting the set of future configurations. As per our initial computational results, such a configuration set results in considerable improvement in part demand forecasts. Currently, we only consider attribute associations which are frequent (e.g. above minimum support or confidence). In the next step, the association rule mining algorithm can be enhanced to look for other relations. Also, in current implementation we simply remove attribute association which are having the conflict with each other or with product configuration rules. As a next step, we will try to develop approaches which can readjust attribute association in case of conflicts. For example, we can have the same selection rate for attributes if they are selected together. Further computational tests are required to validate and improve our assumptions on attribute associations which can improve the selection of future configurations.

REFERENCES

[1] R. Agrawal, T. Imielinski, and A. Swami, 'Mining association rules between sets of items in large databases', in *Proc. 1993 ACM-SIGMOD Int. Conf. Management of Data (SIGMOD-93), Washington DC*, pp. 207–216, (1993).

[2] KimAllan Andersen and Daniele Pretolani, 'Easy cases of probabilistic satisfiability', *Annals of Mathematics and Artificial Intelligence*, **33**(1), 69–91, (2001).

[3] Yves Bastide, Nicolas Pasquier, Rafik Taouil, Gerd Stumme, and Lotfi Lakhal, 'Mining minimal non-redundant association rules using frequent closed itemsets', in *Computational Logic CL 2000*, eds., John Lloyd, Veronica Dahl, Ulrich Furbach, Manfred Kerber, Kung-Kiu Lau, Catuscia Palamidessi, LusMoniz Pereira, Yehoshua Sagiv, and PeterJ. Stuckey, volume 1861 of *Lecture Notes in Computer Science*, 972–986, Springer Berlin Heidelberg, (2000).

[4] Guillaume Bécan, Razieh Behjati, Arnaud Gotlieb, and Mathieu Acher, 'Synthesis of attributed feature models from product descriptions: Foundations', Rapport de Recherche RR-8680, Inria Rennes, (feb 2015).

[5] Guillaume Bécan, Nicolas Sannier, Mathieu Acher, Olivier Barais, Arnaud Blouin, and Benoit Baudry, 'Automating the formalization of product comparison matrices', in *Proceedings of the 29th ACM/IEEE International Conference on Automated Software Engineering*, ASE '14, pp. 433–444, New York, NY, USA, (2014). ACM.

[6] Guillaume Bcan, Mathieu Acher, Benoit Baudry, and SanaBen Nasr, 'Breathing ontological knowledge into feature model synthesis: an empirical study', *Empirical Software Engineering*, 1–48, (2015).

[7] A.K. Choudhary, J.A. Harding, and M.K. Tiwari, 'Data mining in manufacturing: a review based on the kind of knowledge', *Journal of Intelligent Manufacturing*, **20**(5), 501–521, (2009).

[8] Gerard Cornuejols, Milind Dawande, Michel Gamache, Francois Soumis, Gerald Marquis, and Jacques Desrosiers, 'A column generation approach for large-scale aircrew rostering problems', *Oper. Res.*, **47**, 247–262, (February 1999).

[9] Fabio G. Cozman and Lucas Fargoni di Ianni, 'Probabilistic satisfiability and coherence checking through integer programming', *International Journal of Approximate Reasoning*, **58**(0), 57 – 70, (2015). Special Issue of the Twelfth European Conference on Symbolic and Quantitative Approaches to Reasoning with Uncertainty (ECSQARU 2013).

[10] T. Hadzic, S. Sathiamoorthy, R. M. Jensen, H. R. Andersen, J. Møller, and H. Hulgaard, 'Fast backtrack free product configuration using precompiled solution space representations', in *Proceedings of the International Conference on Economic, Technical and Organisational aspects of Product Configuration Systems*, (2004).

[11] Michael Hahsler, Bettina Grün, and Kurt Hornik, 'arules - a computational environment for mining association rules and frequent item sets', *Journal of Statistical Software*, **14**(15), 1–25, (9 2005).

[12] Roberto E. Lopez-Herrejon, Lukas Linsbauer, Jos A. Galindo, Jos A. Parejo, David Benavides, Sergio Segura, and Alexander Egyed, 'An assessment of search-based techniques for reverse engineering feature models', *Journal of Systems and Software*, **103**(0), 353 – 369, (2015).

[13] Efthimia Mavridou, DionisisD. Kehagias, Dimitrios Tzovaras, and George Hassapis, 'Mining affective needs of automotive industry customers for building a mass-customization recommender system', *Journal of Intelligent Manufacturing*, **24**(2), 251–265, (2013).

[14] Rainer Paffrath, 'Mining product configurator data', in *Modern Concepts of the Theory of the Firm*, eds., Gnter Fandel, Uschi Backes-Gellner, Manfred Schlter, and JoergE. Staufenbiel, 110–121, Springer Berlin Heidelberg, (2004).

[15] Tilakraj Singh and Narayan Rangaraj, 'Generation of predictive configurations for production planning', in *Proceedings of the 15th International Configuration Workshop*, eds., Michel Aldanondo and Andreas Falkner, pp. 79–86. CEUR Workshop Proceedings, (2013).

[16] C. Sinz, A. Kaiser, and W. Küchlin, 'Formal methods for the validation of automotive product configuration data', *Artificial Intelligence for Engineering Design, Analysis and Manufacturing*, **17**(1), 75–97, (JAN 2003). Special issue on configuration.

[17] Peter Walley, Renato Pelessoni, and Paolo Vicig, 'Direct algorithms for checking consistency and making inferences from conditional probability assessments', *Journal of Statistical Planning and Inference*, **126**(1), 119 – 151, (2004).