

## **RECO: REPRESENTATION AND EVALUATION OF CONFIGURABLE OFFERS**

Martin Bichler, Jayant R. Kalagnanam, Ho Soo Lee  
*IBM T. J. Watson Research Center, PO Box 218, Yorktown Hts, NY 10598,*  
{bichler, jayant, leehs}@us.ibm.com

**Abstract:** Up until now, electronic negotiations have primarily focused on the trading of simple goods and services, where products can be described either by price alone or as a set of attributes. There is little support for trading complex, configurable products, such as computer systems, automobiles, insurances and services in general. Companies need to communicate offers including complex rules and business policies, and they need decision support to evaluate these offers. This paper describes RECO, a decision support system for the representation and evaluation of configurable offers. Configurable offers allow multiple values for each attribute and they include rules on how to combine the various attribute values and how to price a desired configuration. From a configurable offer, a user can extract offers for individual configurations. RECO provides a compact representation for configurable offers using propositional logic, and helps a user in finding the top L configurations based on her preferences. In a multi-sourcing setting, it provides support for identifying the optimal sourcing strategy subject to considerations such as minimum/maximum number of winners and homogeneity of attributes across bids. We draw on mathematical programming, propositional logic and decision analysis.

**Keywords:** Configurable products, decision analysis, propositional logic, integer programming, XML/EDI

### **1. INTRODUCTION**

Electronic procurement systems have emerged as one of the tools from the B2B era that has found widespread adoption as an integral part of enterprise software (line56.com). One of the primary advantages of such systems is the automation of negotiations between business partners such as

manufacturers and their suppliers using (among other techniques) reverse auctions. A number of systems are now commercially available for supporting electronic negotiations - e.g. Perfect.com, Clarus, SAPMarkets to name a few. However, most systems that support the Request-for-Quote (RFQ) process restrict the nature of the bids that can be represented and communicated between the parties. The assumption in all of the systems available commercially is that offers (or bids) are described as set of attribute value pairs. In practice, however, offerings are more than simple attribute value pairs. Suppliers often provide configurable systems and provide buyers with a series of choice for each of the components. Alternately, if the negotiations are for composite services (such as those required for construction projects), it might be possible to use a Web services infrastructure to compose the required composite service from multiple service providers. The focus of this paper is to provide decision support for configurable offers. In this context, we address two key issues: (i) a representation to capture the configurable offers compactly, and (ii) tools to identify the best few configurations based on the buyer's preferences

Configurable offers are more common than what one may suspect at first. Most services such as insurance or transportation can be considered as configurable offerings. Another example is personal computers. Vendors such as IBM, Dell, and Compaq, describe their offerings along 20 to 30 attributes on their Web site, and each of these attributes can take multiple values. This way, customers can configure and order a PC tailored to their individual needs. For example, suppose a PC has only three attributes, namely processor speed, hard disk size, operating system, and price. A supplier could specify that there are three processors available {850MHz, 950 MHz, and 1GHz}, as well as two sizes of hard disks {10GB and 15GB}. The base configuration (850MHz, 10GB) prices for \$1000. A configuration with a 1GHz computer is \$100 more, and one with a 15GB hard disk costs an additional \$200. Of course, there can be many more rules, which need to be considered. For example, a certain operating system might require a minimal processor speed to function properly, and the supplier might specify a linear decreasing volume discount.

A typical configurable offer consists of a collection of configurations where each configuration specifies a selection for each of the many attributes of the component. A more compact technique for specifying a large set of configurations is to specify the set of attributes and the associated range of values. A set of constraints also needs to be specified that restricts infeasible combinations. This type of a *configurable offer* exhibits some combinatorial aspects. With only 10 attributes and 5 possible attribute values for each of them, there are already 9.7 million possible configurations. An additional

specification that is required is the price associated with each configuration. Given the large number of configurations that is possible, the informational complexity of specifying the entire price function might become prohibitive. If a functional specification of price is possible then this simplifies the specification of the price for each configuration, however this is uncommon in practice. A common pricing rule used is based on “mark ups” as indicated in the PC example. The price for each attribute level is provided as a mark up over the base configuration and this significantly reduces the informational complexity since the price mark up is assumed to be independent of other attributes. However, such a specification is not complete since there are always promotional prices based on selecting attribute bundles or a price premium for difficult or over-demanded configurations. There is a clear need to provide expressive languages that provide a compact representation for configurable bids and communicate these rules in a machine-readable way. In this paper, we present a restricted logical representation (using propositional logic) for capturing feasibility, price markups and promotion rules that appears to capture the requirements in real life settings for configurable products and services.

Given a configurable offer, finding the best configuration/s among all these is not straightforward. Traditional bid evaluation tools can only handle individual configurations described as attribute-value pairs. It is necessary to provide bid evaluation tools that operate on configurable offers and their associated configuration and pricing rules, to identify the top L configurations based on buyer preferences. Another aspect of this paper is a decision support tool that uses a set of integer programming models to identify the top L configurations based on buyer preferences. The configuration rules and price promotions based on attribute bundling represented as propositional logic are incorporated into an integer programming formulation by converting propositional statements into linear inequalities.

In section 2, we discuss the informational complexity of specifying configurable offers and introduce price mark ups with promotional bundles as a practical way of managing this complexity. We introduce the use of propositional logic as a language for capturing configuration and promotional pricing rules in a compact way. We also introduce an XML schema that enables suppliers to express pricing rules for different configurations, volume discounts, as well as constraints among attribute values. Section 3 develops a set of integer programming models to identify the top L configurations in different negotiation settings. A set of models for optimal allocation in a multisourcing context is also presented. The conversion of propositional logic into linear inequalities that can be easily

incorporated into integer programs is discussed. Section 4 provides a discussion of some prototype applications for configurable products and services. Section 5 will conclude with a discussion and comparison to other approaches.

## 2. COMPACT DESCRIPTION OF CONFIGURABLE OFFERS

A simple approach to representing a configurable offer is to allow the specification of a set of attributes and a range of allowable values for each attribute. Therefore any offer  $i$  with  $J$  attributes can be written as  $J$ -tuple where each element is a list of allowable values  $\{v_{ij} \mid j=1,\dots,J; v_{ij} \in V_j\}$  where  $V_j$  is the set of allowable values for attribute  $j$ . Notice that this already allows for a very large number of configurations based on the size of  $V_j$ . However, there still remain two issues regarding representation: (i) representing the price for each allowable configuration, and (ii) representing configuration constraints and price promotions. We discuss each of these issues in the following subsections. Finally, we provide a XML based schema for specifying configurable offers.

### 2.1. Functional specification of price for a configurable offer

Given a configurable offer defined as  $\{v_{ij} \mid j=1,\dots,J; v_{ij} \in V_j\}$  we have two choices for specifying the price for each configuration. The first choice is to enumerate the price for each configuration. However, as we indicated earlier such an enumeration could be exponential and communicating such offers would be difficult. A compact alternative is to force the suppliers to provide a functional description of price as a function of the attributes. We have adopted the later approach but restricted our attention to a specific function form that is most commonly encountered in practice – an additive price function where the impact of each attribute on price is additive. This formalizes the mark up based pricing schemes where a price mark up is associated with each level of an attribute. Notice that the price mark-up for each attribute can be non linear as shown below.

Assuming additivity of the attributes, the total price  $p_i$  for a particular bid/offer  $i$  can be written as:

$$p_i = q_i p_{bi}(q_i) + q_i \sum_j f_{ij}(v_{ij}) \quad (1)$$

where  $q_i$  is quantity,  $p_{bi}(q_i)$  is the base price per item as a function of quantity (i.e. specifies a volume discount), and  $f_{ij}(v_{ij})$  is a functional specification for the impact of particular attribute values  $v_{ij}$  on the price of a product. The individual functions  $p_{bi}(q_i)$  and  $f_{ji}(v_{ij})$  can in general be nonlinear. For any attribute  $j$ , the difference  $f_{ji}(v_{ij}=v_{j,hi}) - f_{ji}(v_{ij} = v_{j,base})$  represents the price markup for level  $hi$  with respect to the *base* level.

Examples would be discrete functions, which specify price markups for different types of CPUs (850 MHz, 900 MHz, 1GHz), as well as continuous functions, which specify the impact of decreasing lead time on price. This functional form can be sent to the auctioneer in an XML-based interchange format. For these purposes, we have designed CPML, an XML schema to describe configurable multi-attribute offers which is discussed in section 2.3.

## 2.2 Configuration Rules and Promotion Rules

It is often essential for suppliers to express rules, which define constraints on the combination of attribute values, or discounts and mark ups based on some combination of attribute values. If we model each attribute level with a boolean variable that indicates whether this level is chosen for a particular configuration, then we can capture configuration constraints using propositional logic sentences. This is a compact representation for rules that are specified on discrete values rather than on ranges. Similarly, promotion pricing based on discrete levels rather than on ranges leads to compact representations using propositional logic sentences. In this paper we have restricted our attention to constraints that can be specified using propositional logic. This appears to be sufficient in the examples that we have examined for both configurable products (such as PCs) and services (such as insurance and logistics).

For instance, a *configuration rule* may include compatibility restrictions, saying attribute value  $x_{112}$  cannot be connected to attribute value  $x_{123}$ , or requirements like attribute value  $x_{112}$  and  $x_{132}$  needs attribute value  $x_{123}$ . Here,  $x_{ijk}$  is a binary indicator variable that takes a value of 1 if the level  $k$  of attribute  $j$  and offer  $i$  is chosen. In this section we are only considering configuration rules within an offer and, hence, omit the subscript  $i$  for the sake of readability. For example, the proposition

$$x_{23} \Rightarrow \neg x_{31} \tag{2}$$

describes the configuration rule that if a certain motherboard,  $x_{23}$ , then the buyer is restricted from using a certain type of CPU,  $x_{31}$ . Logical implication ( $\Rightarrow$ ) allows, that if any other kind of motherboard is selected, the

particular CPU may or may not be chosen. In general, most configuration rules are of the implication variety where the antecedent are propositional sentences.

Another type of rules, which is often found in practice are so called *discount rules*. For example,

$$x_{12} \wedge x_{31} \Leftrightarrow p^- \quad (3)$$

where  $x_{jk}$  again describe particular attribute values and  $p^-$  describes a certain discount (or markup) enforces a discount upon selection of these attribute values. The discount is only given, if and only if  $x_{12} \wedge x_{31}$  is true. If  $x_{12} \wedge x_{31}$  is false, then no discount will be granted. Therefore, we use the equivalence operator ( $\Leftrightarrow$ ) for discount rules.

In order to express these types of constraints we can use propositional logic operators (and  $\wedge$ , or  $\vee$ , implies  $\Rightarrow$ , equivalent  $\Leftrightarrow$ , not  $\neg$ ) (Russell & Norvig, 1995). A rule can then be expressed in terms of logical operations between attribute values (see section 3.4). Propositional logic is sufficient for capturing the configuration constraints and pricing rules considered in this paper.

A prerequisite to evaluate configurable offers is a number of message formats to describe these offers in a compact way. As a first step we will describe the necessary constructs to express configurable offers in a mathematical notation. Based on this we will introduce CPML, an XML schema to describe configurable offers and communicate them as EDI messages over the Internet.

### 2.3. CPML – an XML schema for configurable products

In business-to-business relationships, it is essential to have product data and offers available in machine-readable format. Buyer organizations want to be able to import product data into their ERP (enterprise resource planning) systems, in order to make this data available to employees internally, to compare product specifications of different suppliers, or to reuse product and price data for various planning, engineering or accounting purposes.

Many suppliers make their catalog data available in various formats, such as ANSI X12 832, Ariba cXML<sup>1</sup>, Catalog Interchange Format (CIF), Commerce One xCBL<sup>2</sup>, Requisite eCX 2.0<sup>3</sup>, and RosettaNet PIP 2A1<sup>4</sup>.

<sup>1</sup> <http://www.cxml.org/>

<sup>2</sup> <http://www.xcbl.org/>

<sup>3</sup> <http://www.ecx-xml.org/>

<sup>4</sup> <http://www.rosettanet.org>

Based on the various standards and classification schemes many e-marketplace providers as well as vendors of e-procurement systems provide tools and services for catalogue aggregation. The promise is that buyers can perform searches across multiple suppliers, getting up-to-the-minute pricing, inventory, and sourcing information. In fact, the management of catalogue data is the key feature of most e-procurement solutions. Today's catalogue standards focus mostly on parts catalogues, where the product is described as a set of attributes and eventually a price. An example is CIF, which is a simple comma delimited text format (see Figure 1. CIF example.).

```
DATA
942888711,100,, "Ball-point Pen",1213376,1.95,EA,,,,,
942888711,101,, "No. 2 Pencil",1213377,1.50,DZN,,,,,
942888711,102,, "Rubber Eraser",1213472,0.25,PK,,,,,
942888711,103,, "Stapler, Standard",1237461,2.95,BX,,,,,
ENDOFDATA
```

Figure 1. CIF example.

Currently, there is no support for the description of configurable offers in a machine-readable way. We have defined CPML, an XML schema, which can be used to describe configurable products. The schema defines messages for a product request, for conventional multi-attribute offers, as well as configurable offers.

Figure 2 shows the main elements of a CPML configurable offer element (TAG: ConfigurableOffer) in a tree structure. A configurable offer defines either a single price for a base configuration and quantity, or a volume discount price (TAG: VolumeDiscountPrice), which is a function of quantity. The offer can have multiple attributes, which are either fixed attributes (TAG: FixedAttr), range attributes (TAG: RangeAttr), optional attributes (TAG: OptionalAttr), or discrete attributes (TAG: DiscreteAttr). Fixed attributes are described as simple attribute-value pairs, similar to the ones one can find in traditional catalogue standards. Range attributes allow the specification of a certain range of values and their impact on price. For example, one can specify lead time as an attribute ranging from 7 to 30 days. The faster the good needs to be delivered, the higher the markup on price will be. Optional attributes specify dichotomous features. For example, a PC might be delivered with a certain modem or without. Discrete attributes allow for a larger number of allowed attribute values. With discrete attributes a buyer can specify a number of different modems to choose from, as well as a price markup for each of them.

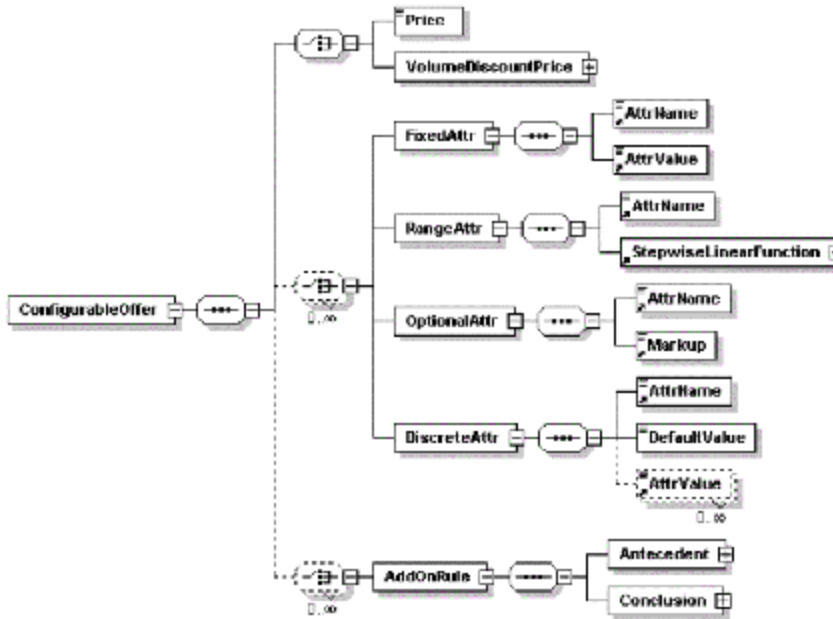


Figure 2. Main elements of CPML configurable offer.

In addition a configurable offer can specify so called add on rules (TAG: AddOnRules). Add on rules have the form of a logical equivalence or an implication with an antecedent and a conclusion. The antecedent can be any logical proposition including conjunctions, disjunctions, negations as well as atomic propositions. The conclusion defines in addition a Markup tag, which specifies a certain discount or markup. This way, suppliers can specify a certain discount as a result of the selected attribute values.

A configurable offer is one of a sequence of messages and a response to a particular request, which specifies a buyer's basic requirements. The tags defined in CPML can be used standalone or they can also be embedded to enrich existing catalogue standards. In the first version of this XML schema we have tried to make it easy to understand and use, but at the same time flexible enough to cover the broad range of business rules, one can find in practice. Using the CPML tags we were able to describe configurable offers in a wide variety of industries ranging from logistics to insurances and electronic components.

### 3. BID EVALUATION FOR CONFIGURABLE OFFERS

The basic purpose of CPML is to communicate pricing rules about configurable offers in a compact way. This data can then be used to evaluate and compare offerings from various suppliers. The RECO bid evaluation methodology provides a suite of algorithms to evaluate and select the best (or top K) out of the multiple possible configurations of a *configurable offer*. It helps the user to score and compare the configurable offers of multiple suppliers. RECO operates directly on the rules and functional specifications of a configurable offer without the need to enumerate all possible configurations. This avoids the computationally expensive process of generating, transmitting and evaluating huge amounts of individual configurations by just communicating the rules for a single configurable offer. RECO consists of two consecutive steps, namely preference elicitation and bid evaluation which is based on integer programming techniques. The treatment of logical configuration rules and a buyer's business rules requires additional consideration and is treated in a separate subsection.

#### 3.1. Preference elicitation

In practical implementations, the elicitation of a buyer's preferences, and consequently the assessment of trade-offs is of pivotal importance. A common approach is based on the use of established decision analysis techniques, such as MAUT (Keeny & Raiffa, 1993), SMART (Edwards, 1977) or AHP (Saaty, 1980). Although advanced versions of MAUT and AHP can model interactions among attributes, the basic techniques use a linear, weighted value function, which assumes *preferential independence* of all attributes. An attribute  $x$  is said to be preferentially independent of  $y$  if preferences for specific outcomes of  $x$  do not depend on the value of attribute  $y$  (Olson, 1995).

We next introduce some terminology and notation. Consider  $I$  bids (or offers) and  $J$  attributes. Each attribute  $j \in J$  has an attribute space of  $K_j$ . A multi-attribute offer, received by the buyer, can then be described as an  $n$ -dimensional vector  $\mathbf{v}_i = (v_{i1} \dots v_{ij})$  where  $v_{ij}$  is the level of attribute  $j$ . In the case of an additive scoring function  $S(\mathbf{v}_i)$  the buyer evaluates each relevant attribute  $v_{ij}$  through a scoring function  $S_j(v_{ij})$ . The overall value  $S(\mathbf{v}_i)$  for a bid  $\mathbf{v}_i$  is given by the sum of all individual scorings of the attributes. It is convenient to scale  $s_i$  and each of the single-attribute utility functions  $S_j(\cdot)$  from zero to one. That is, for a bid  $\mathbf{v}_i$  and a scoring function that has weights  $w_1 \dots w_J$ , the overall utility for a bid is given by

$$s_i = S(\mathbf{v}_i) = \sum_{j \in J} w_j S_j(v_{ij}) \text{ and } \sum_{j \in J} w_j = 1 \quad (4)$$

The problem a buyer faces is to determine appropriate  $S_j(\cdot)$  functions and  $w_j$  weights. An *optimal auction* is allocating the deal to the suppliers in a way that maximizes the utility for the buyer, i.e. to the supplier providing the bid with the highest overall utility score for the buyer. The function  $\max s_i$  with  $I \times I \times I$  gives us the utility score of the winning bid and can be determined through open-cry or sealed-bid auction schemes.

The assessment of appropriate weights  $w_j$  is key to MAUT and is an important aspect of a “good” preference model. Several techniques have been proposed in the traditional decision analysis literature to help users assign reasonable weights. One approach is called *pricing out* because it involves determining the value of one objective in terms of another (e.g. dollars). For example, one might say that 5 days faster delivery time is worth \$400. The idea is to find the indifference point, i.e. determine the marginal rate of substitution between two attributes. Although this concept seems straightforward, it can be a difficult assessment to make.

Since many decision makers feel unable to provide exact weights, some of the more recent approaches only ask for uncertain estimates. For example, methods from fuzzy decision analysis use fuzzy sets for weights and individual scoring functions and fuzzy operators for the aggregation of those fuzzy sets (Ribeiro, 1996). AHP uses a different approach to weight determination. A principle used in AHP is that comparative judgments are applied to construct a symmetric matrix of pair-wise comparisons of all combinations of attributes. The method is based on the mathematical structure of consistent matrices and their associated right-eigenvector’s ability to generate true or approximate weights. The right eigenvector of the matrix results in the weights for the different objectives. More recent approaches try to estimate the buyers preferences based on comparisons of alternatives (Bichler, Lee, Lee, & Chung, 2001; Iyengar, Lee, & Campbell, 2001). These techniques assume weaker decision makers and do not ask for attribute-level utility assessments.

### 3.2 Bid Evaluation

There are two primary scenarios for bid evaluation:

- *Single Sourcing*: The buyer receives a configurable bid from one or more sellers and is interested in identify the top  $K$  configurations based on her preferences. The key assumption for this setting is that all the demand is sourced to a single supplier.

- *Multiple Sourcing*: In this setting the buyer is willing to split the demand across multiple suppliers. However, when buying multiattribute commodities from different suppliers one has to impose additional constraints over the choice. We discuss two common constraints: (i) constraints on the total number of winning suppliers, and (ii) homogeneity constraints that restrict the choice across suppliers to have the same value for one or more attributes.

In the following analysis we restrict ourselves to discrete price markups, where each attribute value  $v_{ij}$  of bid  $i$  (with indicator variable  $x_{ijk}$  for level  $k$ ) has an individual markup  $m_{ijk}$ . In addition, each offer specifies a quantity  $q_i$ . The description of a configurable multi-attribute offer is now formulated as in (5) with  $p_i$  being the unit price of a particular configuration  $\mathbf{X}_i$ . (Note that this is a 0-1 matrix that indicates the levels chosen for each attribute).

$$p_i = p(q_i, \mathbf{X}_i) = p_{bi} + \sum_j \sum_k m_{ijk} x_{ijk} \quad (5)$$

### 3.2.1 Single Sourcing

We first model the simplest setting where the buyer attempts to identify the top  $L$  configurations from a single configurable offer. This model subsequently becomes a kernel that is used in the multiple sourcing setting.

Identifying the best configuration can be modeled as a variation of the *multiple-choice knapsack problem* (Martello & Toth, 1990). We will drop the subscript  $I$  for the binary decision variable  $x_{jk}$  since we consider only one offer in this section. The objective function as written in (6) maximizes the weighted score. Note, that we assume an *additive, quasi-linear utility function* with a linear (decreasing) scoring function on price,  $s_p(p_i)$ . Constraint (7) specifies that for each attribute exactly one value must be selected. Constraint (8) restricts the unit price to be smaller or equal to the buyer's unit reservation price  $C$ . In (9) we determine the value of the continuous variable  $p_i$ . This constraint could also be omitted, so that the price  $p_i$  needs to be determined outside the optimization based on the selected attribute values.

$$\max \sum_j w_j \left( \sum_k s_{jk} x_{jk} \right) + w_p s_p(p_i) \quad (6)$$

subject to

$$\sum_k x_{jk} = 1 \quad \forall j \in J \quad (7)$$

$$\sum_j \sum_k m_{ijk} x_{jk} + p_{bi} \leq C \quad (8)$$

$$\sum_j \sum_k m_{ijk} x_{jk} + p_{bi} = p_i \quad (9)$$

$$x_{jk} \in \{0,1\} \quad \forall j \in J, \forall k \in K \quad (10)$$

The procedure allows suppliers considerably more flexibility in specifying offers, while at the same time, the bids can be ranked and suppliers can compete in an open-cry manner.

Often, a buyer is interested in evaluating the top  $L$  configurations and subsequently choosing an appropriate configuration by examination. In such a setting we need to identify the  $L_{\text{th}}$  best solution. This is not easy in general – it is equivalent to finding the  $L_{\text{th}}$  shortest path in a network (Ahuja, Mananti, & Orlin, 1993). We model the problem by adding additional knapsack cuts to the formulation presented above (equations 7-10) that cut off the top  $L-1$  solutions. Let us denote the  $L_{\text{th}}$  optimal solution by a set of indices  $N_L$  that indicate the levels chosen for each attribute. For example for the optimal solution this set is  $N_1$  and we can eliminate the optimal solution from our search by adding the constraint where  $|N_1|$  is the cardinality of the set.

For 2<sup>nd</sup> best solution add:

$$\sum_j \sum_{k \in N_1} x_{jk} \leq |N_1| - 1 \quad (11)$$

For 3<sup>rd</sup> best solution add:

$$\sum_j \sum_{k \in N_2} x_{jk} \leq |N_2| - 1 \quad (12)$$

For the  $L$ th best solution add:

$$\sum_j \sum_{k \in N_{L-1}} x_{jk} \leq |N_{L-1}| - 1 \quad (13)$$

### 3.2.2. Multiple Sourcing

If the buyer is willing to source her demand (usually specified as an acceptable range  $[D_{min}, D_{max}]$ ) from multiple suppliers then identifying the optimal set of configurations (and the appropriate suppliers) is solved as a two-step procedure. In the first step we select the best possible configuration for each offer based on the buyer's scoring function using the formulation in equation (6-10). In a second step, the resulting "best configurations" are in the form of conventional multi-attribute offers and can be allocated using a knapsack problem as shown below.

$$\max \sum_i (q_i s_i) r_i \quad (14)$$

subject to

$$D_{min} \leq \sum_i q_i r_i \leq D_{max} \quad (15)$$

$$\sum_i q_i p_i r_i \leq C \quad (16)$$

$$r_i \in \{0,1\} \quad \forall i \in I \quad (17)$$

The objective in this optimization is to maximize the overall score (2), where  $s_i$  is the unit score of the optimal configuration from bid  $i$  and  $q_i$  is the quantity of bid  $i$ , so that the supplied quantity satisfies the lower and upper bound for the demand (3). The overall reservation price  $C$  is considered in (4) where  $p_i$  is the unit price of the optimal configuration of bid  $i$ . We introduce the binary decision variables  $r_i$  to indicate the bids selected by the buyer (6).

In real-world settings there are several considerations besides cost minimization. These considerations are specified as a set of constraints that need to be satisfied while selecting a set of winning bids. We discuss two such rules, which have been shown to be relevant in practical applications.

- *Number of Winning Bidders*: An important multiple sourcing consideration is the number of winners. On the one hand, buyers want to make sure that the entire supply is not sourced from too few suppliers, since this creates a high exposure if some of them are not able to deliver on their promise. On the other hand, having too many suppliers creates a high overhead cost in terms of managing a large number of supplier relationships. These considerations introduce constraints on the

minimum,  $O_{min}$ , and maximum,  $O_{max}$ , number of winning suppliers in the solution to the winner determination problem.

- *Homogeneity of the Purchase*: A basic problem, which arises from multi-attribute auctions in the case of multiple sourcing is the heterogeneity of goods purchased from different suppliers. The cost minimizing solution might be one where the bids of all winning suppliers have different values in all attributes. While this is not necessarily a problem for all attributes, it can be important in certain applications to enforce homogeneity of a certain attribute in the set of winning bids.

Incorporating these constraints into the formulation introduces complicating constraints that requires merging the two-step procedure above into a single albeit large formulation. Equations (18) – (26) show the overall MIP model for the evaluation of configurable multi-attribute offers considering multiple sourcing and homogeneity constraints.

$$\max \sum_i q_i \left\{ \sum_j w_j \left( \sum_k s_{ijk} x_{ijk} \right) + w_p s_p \sum_j \sum_k m_{ijk} x_{ijk} + w_p (s_p p_{bi} + d) y_i \right\} \quad (18)$$

subject to

$$\sum_k x_{ijk} = y_i \quad \forall j \in J, \forall i \in I \quad (19)$$

$$\sum_i q_i \left( \sum_j \sum_k m_{ijk} x_{ijk} + p_{bi} y_i \right) \leq C \quad (20)$$

$$D_{min} \leq \sum_i q_i y_i \leq D_{max} \quad (21)$$

$$O_{min} \leq \sum_i y_i \leq O_{max} \quad (22)$$

$$\sum_{i \in T_{jk}} x_{ijk} \leq \|T_{jk}\| z_{jk} \quad \forall j \in J \quad (23)$$

$$\sum_k z_{jk} = 1 \quad \forall j \in J \quad (24)$$

$$x_{ijk} \in \{0,1\} \quad \forall i \in I, \forall j \in J, \forall k \in K \quad (25)$$

$$y_i \in \{0,1\} \quad \forall i \in I \quad (26)$$

The objective maximizes the overall score of the selected configurations. As in (15) we assume an additive and quasi-linear scoring function with a linearly decreasing function on price. The variable  $s_{ijk}$  again denotes the score for a particular qualitative attribute value, whereas  $x_{ijk}$  is a binary indicator variable which indicates whether a particular attribute value has been chosen. The variable  $s_p$  describes the slope of the linear scoring function for price, whereas  $d$  is its intercept.

In (19) we select exactly one attribute value for each attribute in an offer, and introduce  $y_i$  as an indicator variable for a particular offer. The constraint in (20) specifies a reservation price  $C$ . (21) restricts the quantity to match an upper and lower bound ( $D_{min}$  and  $D_{max}$ ) specified by the buyer, and (22) limits the number of winners. Finally, (23) and (24) specify homogeneity constraints. In (23) we introduce the indicator variable  $z_{jk}$  that assumes the value 1 if any suppliers are chosen with a bid at level  $k$  for attribute  $j$ . Note that a disaggregated form of this constraint as shown below (23a) provides a relaxation that is stronger.

$$z_{jk} \geq x_{ijk} \quad \forall i, j, k \quad (23a)$$

$T_{jk}$  is defined as the set of bids at level  $K$  for attribute  $J$ . Compared to formula (2) – (6) we have dropped the index  $l$ , because we assume every bidder to submit only one configurable offer.

### 3.2.3 Treatment of logical configuration and discount rules

In this section we discuss how the configuration rules and discount rules are handled within the bid evaluation procedure. The main observation that lies under the approach adopted here is that propositional sentences can be written as linear inequalities with binary variables. Once we convert the logic rules into linear inequalities, these can be added to the integer programming formulations provided above.

We use  $x_{jk}$  to denote the *logical* as well as the *binary variable* in the MIP formulation. For ease of reading we omit the first subscript  $i$  for bids in the first part of this section.

For the evaluation of a configurable offering, these additional rules have to be considered in the IP formulation. In order to obtain an equivalent mathematical representation for any propositional logic expression, one must first consider basic logical operators to determine how each can be transformed into an equivalent representation in the form of an equation or inequality. Raman and Grossman (Raman & Grossmann, 1991) specify transformations, which can then be used to convert general logical

expressions into an equivalent mathematical representation. Some of these transformations are described in Table 1.

Table 1. Representation of logical relations with linear inequalities.

Logical relation	Pure logical expression	Representation as linear inequalities
Logical “OR”	$x_{1l} \vee x_{2l} \vee \dots \vee x_{nl}$	$x_{1l} + x_{2l} \dots + x_{nl} \geq 1$
Logical “AND”	$x_{1l} \wedge x_{2l} \wedge \dots \wedge x_{nl}$	$x_{1l} \geq 1; x_{2l} \geq 1; \dots; x_{nl} \geq 1$
Implication ( $\Rightarrow$ )	$\neg x_{1l} \vee x_{2l}$	$1 - x_{1l} + x_{2l} \geq 1$
Equivalence ( $\Leftrightarrow$ )	$(\neg x_{1l} \vee x_{2l}) \wedge (\neg x_{2l} \vee x_{1l})$	$x_{1l} - x_{2l} \leq 0; x_{2l} - x_{1l} \leq 0$

A common approach to convert a general logical expression into inequalities is to first transform it in its equivalent conjunctive normal form (CNF) representation. CNF involves the application of pure logical operations (and  $\wedge$ , or  $\vee$ , not  $\neg$ ), and is a conjunction of clauses. A clause is defined as a set of basic literals separated by  $\vee$ -operators, such as

$$(x_{12} \vee x_{23}) \wedge (\neg x_{34} \vee x_{45}) \quad (27)$$

CNF can then be expressed as a set of linear inequality constraints, as shown in Table 1. We have chosen this approach to transform the configuration and discount rules in CPML into appropriate constraints in our IP formulation described in (18) – (26). Formulas (28) to (33) show how the proposition in (28) can be translated into linear constraints in our IP formulation.

$$x_{12} \wedge x_{23} \Leftrightarrow p^- \quad (28)$$

$$(\neg(x_{12} \wedge x_{23}) \vee p^-) \wedge (0p^- \vee (x_{12} \wedge x_{23})) \quad (29)$$

$$(\neg x_{12} \vee \neg x_{23} \vee p^-) \wedge (0p^- \vee x_{12}) \wedge (\neg p^- \vee x_{23}) \quad (30)$$

$$x_{12} + x_{23} - p^- \leq 1 \quad (31)$$

$$x_{12} - p^- \geq 0 \quad (32)$$

$$x_{23} - p^- \geq 0 \quad (33)$$

In (28) the equivalence operator has been transformed into a proposition with pure logic operators. Using DeMorgan’s Theorem the negation operator of the first term in brackets is moved inwards, so that we get CNF in (30). Finally, in (31) to (33) CNF is translated into inequalities, which can be added to the integer programming formulation. In addition, we have to

introduce an additional binary indicator variable for  $p^-$  in our model, which indicates the discount if the rule takes effect.

Table 2. Translation of typical configuration and discount rules.

Logical expression	Equivalent linear inequalities
$\wedge x_{ijk} \Rightarrow x_{irs}$ and $i \in I, j, r \in J, k \in K_j, s \in K_r,$ $\forall j \neq r$	$\sum_{ijk \in R} (1 - x_{ijk}) + x_{irs} \geq 1 \quad \forall i \in I$
$\vee x_{ijk} \Rightarrow x_{irs}$ and $i \in I, j, r \in J, k \in K_j, s \in K_r,$ $\forall j \neq r$	$x_{irs} - x_{ijk} \geq 0 \quad \forall ijk \in R, \forall i \in I$
$\wedge x_{ijk} \Leftrightarrow p^-$ and $i \in I, j \in J, k \in K_j$	$\sum_{ijk \in R} (1 - x_{ijk}) + p \geq 1 \quad \forall i \in I$
$\vee x_{ijk} \Leftrightarrow p^-$ and $i \in I, j \in J, k \in K_j$	$x_{ijk} - p \geq 0 \quad \forall ijk \in R, \forall i \in I$
	$\sum_{ijk \in R} x_{ijk} - p \geq 0 \quad \forall i \in I$
	$p - x_{ijk} \geq 0 \quad \forall ijk \in R, \forall i \in I$

The logical expressions in Table 2 describe common forms of configuration and discount rules with only conjunctions or only disjunctions in the antecedent and one literal in the consequent. We have used the notation with three subscripts so that the additional constraints can be added to the optimization formulation in (18) – (26).  $R$  is defined as the set of attribute values in the antecedent of a rule in an offer. Of course, the antecedent and the consequent of these rules can in general be any combination of conjunctions and disjunctions. In other words, with the relations given in Table 1 one can systematically model an arbitrary propositional logic expression as a set of linear equality and inequality constraints.

### 3.2.4 Computational Issues

From a computational point of view the allocation of configurable offers *without homogeneity* constraints is considerably easier to solve than the *problem with homogeneity constraints* described in section 3.2.2. Without homogeneity constraints, the overall winner determination can be split in several smaller problems (see section 3.2.1), in which the best possible configuration for each configurable offer is selected based on a buyer's scoring function. In our numerical simulations, the selection of the best configuration for an offer with four configuration rules, and ten attributes with four attribute values each could find the best configuration in the order

of milliseconds using a commercial optimization package. The results of these individual selection problems are then used in the overall winner determination described in equations (2) – (9).

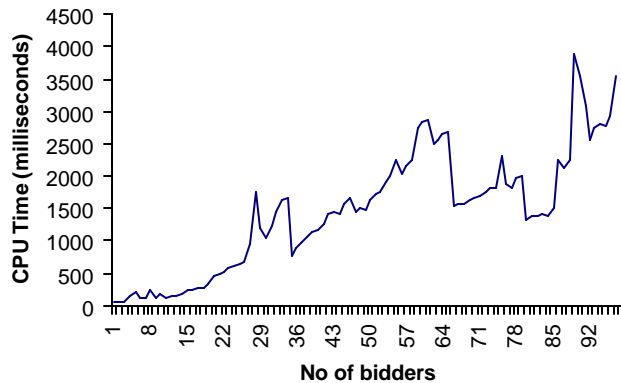


Figure 3. Experimental results for the allocation of configurable offers with increasing numbers of bidders.

The winner determination problem is considerably harder to solve in the presence of homogeneity constraints, because all bids have to be considered at the same time. Figure 3 shows the CPU times of a randomly generated problem instance with 30 attributes, a single homogeneity constraint on one of the attributes, and an increasing number of bidders. Figure 4 investigates the impact of the homogeneity constraints on the runtime of the winner determination. The problem size was constant with 60 bids and 20 attributes and no other side constraints were set for the experiment.

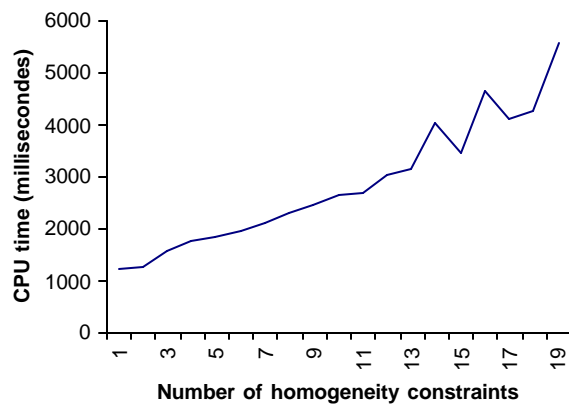


Figure 4. Experimental results for the allocation of configurable offers with increasing numbers of homogeneity constraints.

In our future research, we plan to extend the analysis towards configurable offers, which allow the specification of volume discounts. This aspect adds an additional degree of flexibility to suppliers, however, again at the expense of complexity in the winner determination. RECO has been implemented in Java. It parses CPML documents, transforms them in the appropriate optimisation model and solves the model using IBM's OSL, an optimisation package. The following section describes the integration of RECO in the eHub a marketplace infrastructure based on Web service standards.

#### 4. RECO APPLICATION SCENARIO

Personal computers are an example for physical goods where configurable offerings are in wide-spread use. PC vendors such as Dell, HP, or IBM provide offerings on their web site, where they allow end consumers to select different attribute values for the various attributes of a PC system.

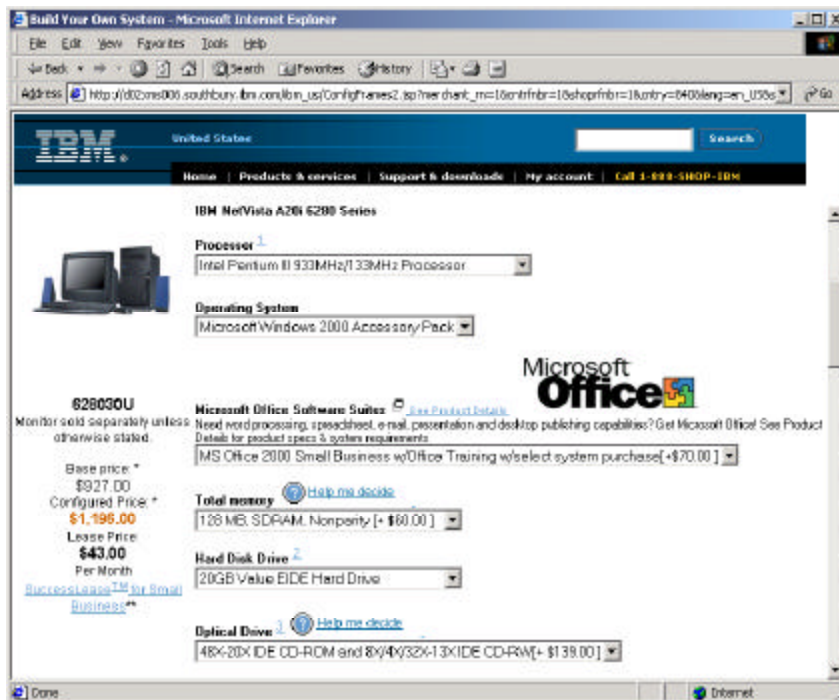


Figure 5. Web-based PC configuration.

Figure 5 shows an example of such a Web-based PC configuration. Users can select different attribute values associated with different mark-up prices. We have chosen the purchasing of PCs as a demonstration scenario for a prototypical catalog aggregation system called eHub. eHub is a tool developed at IBM Research to help suppliers aggregate information from multiple WebServices. We have integrated RECO as a prototypical decision analysis tool in this context to help users evaluate the configurable offerings of WebServices from PC vendors.

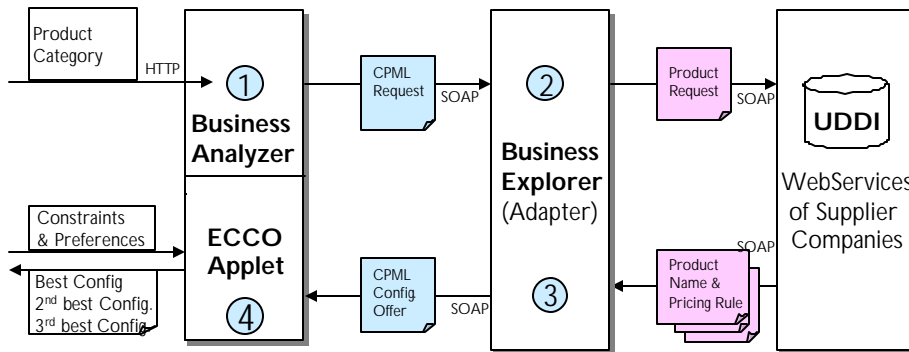


Figure 6. eHub service request and architecture.

Figure 6 provides an overall picture of an eHub service request. eHub consists of two basic components, Business Analyzer and Business Explorer. Business Analyzer provides a number of analysis features needed in a Web service environment. Business Explorer basically provides adapters for the specific Web services of different companies. This is useful, since the responses of the various WebServices are often not standardized and use a different ontology to describe their services.

In a first step, a user enters the product category or type of PC to trigger a CPML request in Business Analyzer (1). Business Analyzer uses an industry specific request template and sends a service request via the SOAP protocol to the Business Explorer WebService (2). The Web services of different companies are not necessarily standardized. Therefore, Business Explorer acts as an adapter and translates the service request into a company specific request, and converts the responses and company specific pricing rules into a configurable offer in CPML(3). Upon initialisation, the RECO applet sends a SOAP request to the Business Explorer to dynamically download the CPML offers for a particular service request. RECO parses the XML documents and reads the pricing rules into an internal object representation. Users can specify their preferences for the various attributes, and RECO generates a list of the individual configurations, which best satisfy the user's preferences. In a demo scenario with only two configurable offers, RECO

detected already more than seventy thousand possible configurations. In our example, RECO returns a list of the 10 best individual configurations for each offer and sorts all of them by score (see Figure 7).

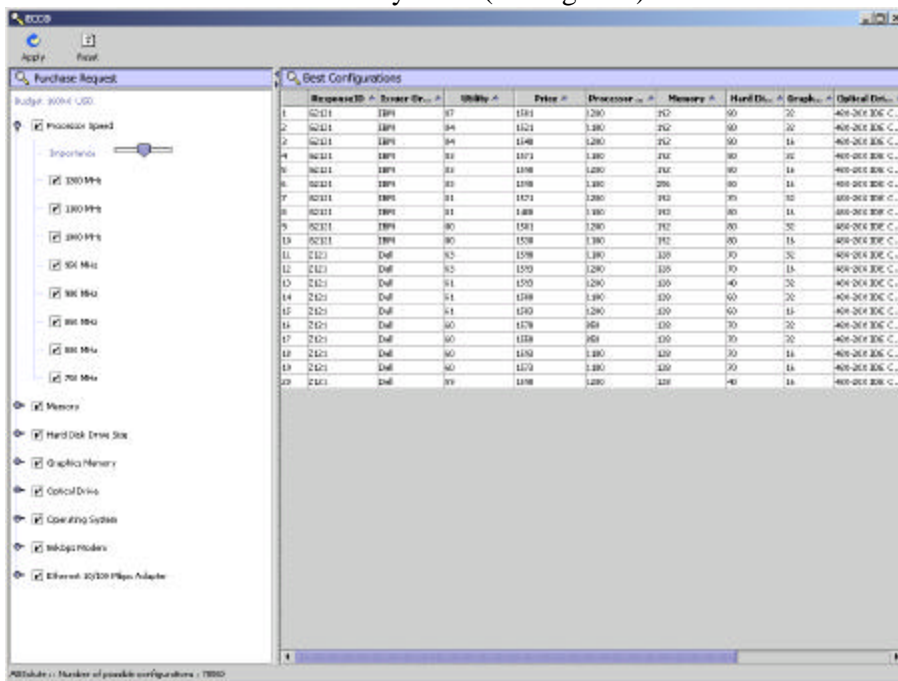


Figure 7. Example of result set.

## 5. CONCLUSION

Until now, most electronic commerce systems have focused on trading simple products and services. Only recently, researchers have focused on complex products and services. Most of these approaches are limited to simple multi-attribute bids, and do not tackle the specific requirements of complex, configurable products. However, a number of newer approaches try to capture more complex business rules. For example, Iwaihara (Iwaihara, 2000) introduces so called dynamic constraints to model complex negotiation conditions, and a query language to select these constrained offerings from a database. Reeves et al. (Reeves, Grosf, & Wellman, 2000) present an approach where offers in a negotiation are expressed in first-order logic. Although, this approach offers greater flexibility, the description of configurable offers becomes more difficult for businesses. Also, no bid evaluation methodology has been suggested for these offers. In contrast, the

goal of CPML was to provide a set of constructs which is on the one hand powerful enough to express real-world offers, but also easy to understand for businesses. CPML uses functional specifications combined with logical propositions to specify and communicate the required pricing and configuration rules. RECO provides a bid evaluation methodology to select one out of the huge number of possible configurations, based on a user's preferences. The methodology has been implemented in eHub, a marketplace infrastructure based on WebService standards. It is planned to deploy this infrastructure in a variety of real-world electronic marketplaces.

## ACKNOWLEDGEMENTS

The authors would like to thank Juhnyoung Lee, Sudhir Verma, Liang-Jie Zhang, and Henry Chang for their contributions to the project.

## REFERENCES

- Ahuja, R. K., Mananti, T. L., & Orlin, J. B. (1993). *Network Flows: Theory, Algorithms, and Applications*. Prentice Hall.
- Bichler, M., Lee, J., Lee, H. S., & Chung, J.-Y. (2001). *ABSolute: An Intelligent Decision Making Framework for E-Sourcing*. Paper presented at the 3rd International Workshop on Advanced Issues of E-Commerce and Web-Based Information Systems, San Jose, CA.
- Edwards, W. (1977). How to use multiattribute utility measurement for social decisionmaking. *IEEE Transactions on Systems, Man, and Cybernetics SMC*, 7(5), 326-340.
- Iwaihara, M. (2000). *Supporting Dynamic Constraints for Commerce Negotiations*. Paper presented at the Second International Workshop on Advanced Issues of E-Commerce and Web-Based Information Systems (WECWIS), Milpitas, CA.
- Iyengar, V. S., Lee, J., & Campbell, M. (2001). *Q-Eval: Evaluating Multiple Attribute Items Using Queries* (Research Report ). New York: IBM Research.
- Keeny, R. L., & Raiffa, H. (1993). *Decision Making with Multiple Objectives: Preferences and Value Tradeoffs*. Cambridge, UK: Cambridge University Press.
- Martello, S., & Toth, P. (1990). *Knapsack Problems*. Chichester, New York: John Wiley & Sons.
- Olson, D. L. (1995). *Decision Aids for Selection Problems*. New York, et al.: Springer.
- Raman, R., & Grossmann, I. E. (1991). Relation between MILP modelling and logical inference for process synthesis. *Computers and chemical engineering*, 15(2), 73-84.
- Reeves, D., Grosz, B., & Wellman, M. (2000). *Automated Negotiations from Declarative Contract Descriptions*. Paper presented at the AAAI-2000 Workshop on Knowledge-Based Electronic Markets, Austin, TX.
- Ribeiro, R. A. (1996). Fuzzy multiple attribute decision making: A review and new preference elicitation techniques. *Fuzzy Sets and Systems*, 78, 155-181.
- Russell, S. J., & Norvig, P. (1995). *Artificial Intelligence - A Modern Approach*. Upper Saddle River, New Jersey, USA: Prentice Hall.
- Saaty, T. L. (1980). *The Analytic Hierarchy Process*. New York, USA: McGraw Hill.

## APPENDIX: ABBREVIATIONS

$C$	unit reservation price
$D_{min}$	lower bound for a buyer's demand
$D_{max}$	upper bound for a buyer's demand
$f_{ij}(v_{ij})$	defines the price markup for a particular attribute value $v_{ij}$ of offer $I$
$I$	set of offers
$J$	set of attributes
$K_j$	set of attribute values for attribute $j$
$L$	set of configurations
$m_{ijk}$	markup of level $k$ of attribute $j$
$N_L$	set of indices for the $L$ th best solution
$O_{min}$	maximum number of winning suppliers
$O_{max}$	maximum number of winning suppliers
$p_{bi}(q_i)$	price of a base configuration for an offer as a function of quantity
$p_i$	total price for a particular offer $i$
$p^-$	a discount or markup specified in logical propositions about configuration rules
$q_i$	total quantity for a particular offer $i$
$s_i$	score of offer $i$
$S_j(\cdot)$	scoring function for attribute $j$
$S(v_j)$	set of attribute values for attribute $j$
$s_p(p_i)$	linear (decreasing) scoring function on price
$T_{jk}$	set of all bids at level $K$ for attribute $J$
$V_j$	set of allowable values for attribute $j$
$v_{ij}$	value of attribute $j$ of offer $i$
$w_j$	weight of attribute $j$
$x_{ijk}$	a binary indicator variable for a particular attribute level $v_{ij}$
$y_i$	an indicator variable that one particular offer has been chosen
$z_{jk}$	indicator variable that assumes the value 1 if any suppliers are chosen with a bid at level $k$ for attribute $j$
$r_i$	best configuration from offer $i$ based on a given scoring function