DIFFERENT STRATEGIES FOR RESOLVING PRICE DISCOUNT COLLISIONS

Henrik Stormer
University of Fribourg
Department for Informatics
Bd de Perolles 90
CH-1700 Fribourg

Keywords: Prices, discount collision, dTree, graphical discount modelling.

Abstract: Managing Discounts is an essential task for all business applications. Discounts are used for turning prospects to new customers and for customer retention. However, if more than one discount type can be applied, a collision may arise. An example is a promotional discount and a discount for a customer. The system has to decide which discount should be applied, eventually also combinations are possible. A bad collision strategy can lead to annoyed customers as they do not get the price that they think is correct.
This paper examines three different business applications designed for small and medium sized companies and shows the applied strategies for resolving discount collisions. Afterwards, a new way of defining discounts, called discount tree (dTree) is introduced. It is shown that with dTrees, collisions can be detected directly when the discounts are defined. A detected collision can then resolved by the administrator.

1 INTRODUCTION

One of the most important decision making factors when buying a good or service is the price (Gijsbrechts, 1993). Therefore, a major task for shops or companies that sell goods or services is to find a price for a customer. Electronic business offers new chances to improve this task because prices can be changed instantaneously on the internet (Jap and Naik, 2004).

A price can be defined as the value agreed upon by the the buyer and the supplier in exchange (Nagle and Holden, 2002). In this paper, we will calculate the price of a product (we will use the term product to specify a good or service) by utilizing a standard price and a discount. A discount should motivate the user to buy the product directly. Additionally, a discount can encourage a customer to buy more and therefore rise the sales volume of the seller.

It is difficult to find a good base price and a discount for products and customers. There exist a number of different strategies (Inoue et al., 2001). Throughout this paper, we will assume that a strategy has been chosen and that a discount has to be applied in a concrete business application. A problem arises, if more than one discount is defined for the same product and customer. The business application has to find one (or a combination) discount. This is called a discount collision.

Discounts are defined by a number of parameters (compare to (Kelkar et al., 2002)):

Customer: If the discount is calculated for the customer an individual price is generated. Typically, this kind of discount is either used for high priced products or for products where the price is negotiated. Sometimes, a discount can be calculated automatically when the customer is classified in a special customer segment.

Quantity: Discounts that depend on the quantity are common (often called scale price). The economic reason is that the marginal costs are decreasing. A typical discount on quantity is defined in a table, where for each quantity interval a discount is assigned.

Time: A time interval can be used to specify the validity of a discount, this is sometimes called promotional. A discount is given for a product only for a period of time, for example one week.

Region: Sometimes, different discounts are applied when the product is sold in different regions. The reason is the buying power that usually varies from region to region.

Other Products: Some vendors package products to bundles (Bakos and Brynjolfsson, 1999; Bakos and Brynjolfsson, 2000). If a customer buys such a bundle, he gets a discount compared to the sum of the individual prices.

Most important, typical discounts are defined by using a combination of the parameters above. A combination of parameters typically lead to the following different discounts:

customer discount: This discount is based on the customer parameter. It is often possible to extend it by defining a time interval and a scale price.

promotional discount: A promotional discount is based on the time paramter. Like the customer discount, this can be extended by defining defining a scale price.

standard discount: Usually, this discount can only be defined by a scale price.

bundle discount: This discount is applied by choosing different products and giving this bundle a special price. Afterwards, the bundle is internally managed as a single product (and therefore, all discounts above can be used).

To make defining discounts easier, some systems offer extended approaches:

price lists: A price list defines list prices for a number of products. Typically, systems offer a way to define price lists for different customer segments. This approach is a simpler way for specifying customer discounts.

product groups: Some systems offer a way to group products, for example by arranging them into categories. Using this approach, a discount is not only applied for one product but for a product group.

Whenever multiple discounts are defined for a product, a discount collision can arise. An example could be customer Smith who wants to buy 5 products X. The standard price for product X is $10. Therefore, without a discount Smith has to pay $50. Let's assume that promotional discount using the scale price parameter is defined for product X:

Quantity	**Price**
1-3	no discount
4-6	-$2 per item

With this scale price, Smith has to pay only $40.

Let's also assume that Smith has a customer discount for product X of $9 per item. Then, with this discount, he has to pay $45. The collision arises between the discount of the customer and the promotional discount. There are now three different possiblities:

- The customer has to pay $40 because of the promotional discount.
- The customer has to pay $45 because of his customer discount.
- The customer has to pay $35. The customer discount defines a price per item of $9 and the scale price of the promotional discount gives $2 off.

2 MODELLING DISCOUNTS GRAPHICALLY

This section shows a new way of defining discounts using a graphical tree, called the dTree (discount tree). Please note that the tree is only used for illustration and graphical modelling purposes, the calculation is done more efficiently (cf 2.3).

2.1 The dTree Composition

This section shows the basic idea behind dTree. A discount is usually defined for one product (it is possible to extend this to product groups). Therefore, the root of the dTree is the product X for which a discount should be defined. For each supported discount parameter, the dTree defines one level. The nodes of the tree consists of elements. An element is an atomic definition of a discount parameter. All elements of one layer define the complete discount parameter set. The root node is *product X*. Level 1 of the tree is the customer parameter. An atomic element in this level is one customer, ie *Customer A*. For each customer, an element is created. This is usually a finite set. Level 2 of the tree is the quantity parameter. For each possible quantity (1, 2, 3,...), one element is created. This set is infinite because a customer can buy whatever quanitity he wants to buy of product X. Level 3 of the tree shows the time. An element of this parameter is one day (ie 21. May 1999). This is also an infinite set.

To make the example easier, all infinite sets are restricted to finite ones by defining upper (and lower) bounds. The quanitity is restricted to 3 (a customer cannot buy more than three product X at once). The time is restricted from the 21.05.99 to 23.05.99 (it is not possible to define discounts with time restrictions for other days). It is also assumed that our shop has only three customers (A,B,C). Then a discount can be defined by connecting one element of each layer. This is a path in the tree from the root to the leaf. A possible path is Product X, Customer B, 1, 23.05, shown in figure 1(a).

A path in the tree defines exactly one discount. Therefore, a discount value can be assigned (for example 10% off). The path of figure one describes the discount: Customer B gets 10% off of Product X if he buys exactly 1 product X at 23. May.

The main advantage of the paths are that it is very easy to define when a collision occurs:

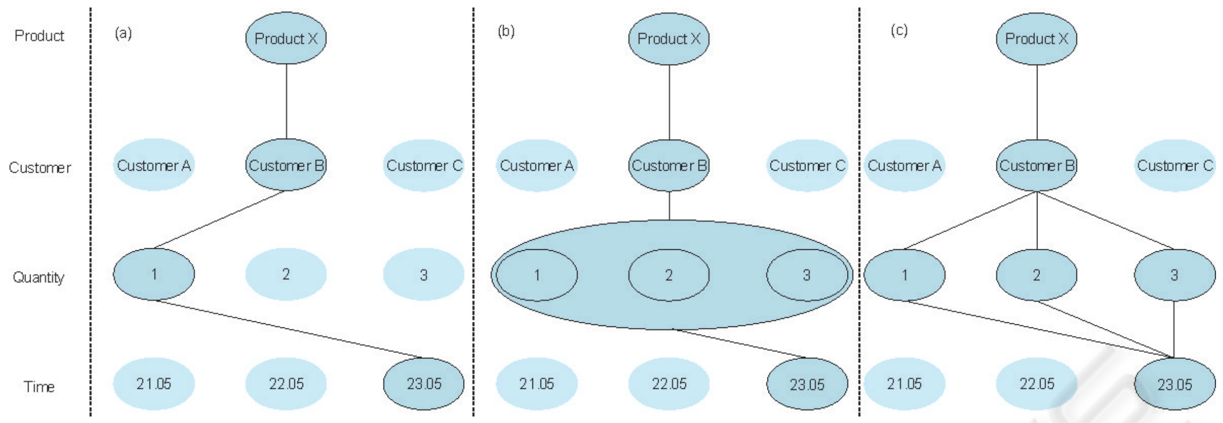

Figure 1: Different illustrations of the dTree.

Theorem 1. *A discount collision between two discounts A and B occurs exactly when they both have the same path.*

Assume that two paths are not equal but a collision occurs. This would be a contradiction of the atomic elements definition. The next section will show how we can group elements for making discount definitions simpler.

2.2 Element Groups

Right now, it is cumbersome to define discounts because all parameters have to be defined exactly. Additionally, it is often the case that only a subset of the parameters should be defined. For example, promotional discounts are usually not customer dependent. Therefore, we introduce the idea of element groups. In a first step, we sort the elements of each layer. The quantity is sorted ascending (1,2,3,...). The same is done for the time (23.05, 24.05, 25.05,...). The customer could be sorted using different attributes, depending on how customer groups (or customer segments) are defined. An element quantity group could be 1 to 3, written as [1,3]. An element group can now be seen as a combination of paths. Suppose that the example of figure 1(a) should not be restricted to quantity 1 but for quantities [1,3], resulting in a path Product X, Customer B, [1,3], 23.05. Then, the tree can be seen as in figure 1(b) where the quantity group is defined. What exactly does an element group define?

Theorem 2. *An element group definition $[a_1, a_k]$ that is used in a path $p = e_1, e_2, e_i, [a_1, a_k], e_{i+2}, e_n$ is only a short description for defining paths for all elements inside the group definition $[a_1, a_k]$:*

$$\begin{aligned} p_1 &= e_1, e_2, e_i, a_1, e_{i+2}, e_n \\ p_2 &= e_1, e_2, e_i, a_2, e_{i+2}, e_n \\ &\vdots \\ p_k &= e_1, e_2, e_i, a_k, e_{i+2}, e_n \end{aligned}$$

This can be seen directly because of the combination of elements to groups. The example of figure 1(b) can be transformed to the three paths (cf. figure 1(c)):

Product X, Customer B, 1, 23.05
Product X, Customer B, 2, 23.05
Product X, Customer B, 3, 23.05

With element groups, it is easy to define discounts that are not restricted to a discount parameter. If, for example, a promotional discount should be created that is not restricted to a customer, it is defined for all customers by using the element group [Customer A, Customer Z] (assume that Customer A is the first and Customer Z the last customer of our sorted customer set. We define the star symbol (*) as a short for all elements: [*].

2.3 Calculating Discount Collisions Efficiently

As we noted earlier, the tree is only used for illustration purposes. It is expensive to add all the paths to the tree. And it is also not necessary:

Theorem 3. *If two paths $p_1 = a_1, \ldots, a_n$ and $p_2 = b_1, \ldots, b_n$ possibly containing element groups are to be checked for a collision, on each layer i three different cases can occur:*

1. *a_i and b_i are both no element group.*

2. *One of the elements, either a_i or b_i is an element group. We assume that a_i is the group (if not, we swap a_i and b_i).*
3. *a_i and b_i are both element groups.*

A collision occurs when for all layers n with $1 \leq i \leq n$ the following tests do not return **no collsion***:*

Case 1: *if a_i and b_i are not equal return* **no collsion**.

Case 2: *if b_i is not an element of a_i return* **no collsion**.

Case 3: *if a_i and b_i do not overlap return* **no collsion**.

The theorem follows from theorem 1 and 2. Theorem 1 says that we have a collision exactly when the path is equal. Therefore, if, in case 1, the two elements are not equal, there is no collision. Theorem 2 says that an element group can be transformed into a number of paths. For case 2, if the element b_i is not inside the element group, the paths do not share and therefore we have no collision. The same is true for case three.

3 DISCUSSION AND OUTLOOK

This paper has shown different ways of defining discounts used in popular business software systems today. It was shown that the automatic calculation of discounts is difficult and can lead to annoyed customers. Afterwards, the paper presented the dTree to define discounts graphically. It was shown that with dTrees, discounts can be defined more easily and for all collisions, the administrator can choose one path.

The dTree is not restricted to the parameters used as an example in this paper, although they are the most popular. It could be extended by using other parameters, for example the shipping region (compare to section 1). Another interesting extension is to show the dTree also to the customers to make the calculation of discounts more transparent.

To further prove the concept, the dTree approach is currently implemented in the eSarine business application system (Werro et al., 2004). eSarine is open source and can be used to manage, offer and sell products on the internet.

REFERENCES

Bakos, Y. and Brynjolfsson, E. (1999). Bundling information goods: Pricing, profits and efficiency. *Management Science*, 45(12).

Bakos, Y. and Brynjolfsson, E. (2000). Bundling and competition on the internet. *Marketing Science*, 19(1).

Gijsbrechts, E. (1993). Prices and pricing research in consumer marketing: Some recent developments. *International Journal of Research in Marketing*, 10.

Inoue, K., Nakajima, H., and Yoshikawa, N. (2001). Price strategies in the e-business age. *NRI Papers*, 23.

Jap, S. D. and Naik, P. A. (2004). Introduction to the special issue on online pricing. *Journal of Interactive Marketing*, 18(4).

Kelkar, O., Leukel, J., and Schmitz, V. (2002). Price modelling in standards for electronic product catalogs based on xml. In *Proceedings of the eleventh international conference on World Wide Web, Honolulu, Hawaii.*

Nagle, T. T. and Holden, R. (2002). *The Strategy and Tactics of Pricing: A Guide to Profitable Decision Making*. Prentice Hall.

Werro, N., Stormer, H., Frauchiger, D., and Andreas, M. (2004). esarine — a struts-based webshop for small and medium-sized enterprises. In *Proceedings of the Workshop Information Systems in E-Business and E-Government (EMISA)*.