Searching optimal product bundles by means of GA-based Engine and Market Basket Analysis

Cosimo Birtolo*, Diego De Chiara*, Simona Losito*, Pierluigi Ritrovato§‖, and Mario Veniero§

* *Poste Italiane S.p.A. – Information Technology*
S – FSTI - Centro Ricerca
80133 Napoli, Italy
{*birtoloc,dechia22,lositosi*}*@posteitaliane.it*

§ *CRMPA – Centro di Ricerca in Matematica Pura ed Applicata c/o DIEII*
Department of Electronic Engineering and Computer Engineering
University of Salerno – 84084 Fisciano, Italy
mveniero@crmpa.unisa.it

‖ *MOMA S.p.A.*
84081 Baronissi, Italy
ritrovato@momanet.it

***Abstract*—The product bundling problem is a challenging task in the e-Commerce domain. We propose a generative engine in order to find the bundle of products that best satisfies user requirements and, at the same time, seller needs such as the minimization of the dead stocks and the maximization of net income. The proposed system named Intelligent Bundle Suggestion and Generation (IBSAG) is designed in order to satisfy these requirements. Market Basket Analysis supports the system in user requirement elicitation task. Experimental results prove the ability of system in finding the optimal trade-off between different and conflicting constraints.**

***Keywords*-Bundle generator; Product bundling; e-Commerce; Association rules; Market Basket Analysis; Genetic Algorithms**

I. Introduction

European online retail sales increased of 11% from 2010 to 2011. Moreover, it is expected that the number of online buyers in Europe will grow from 157 million to 205 million before 2015 (source: Forrester Research - *European Online Retail Forecast, 2010 To 2015*). According to *comScore, Inc*, the United States' retail e-commerce spending totalled $32.9 billion in the second quarter of 2010. Tab.I shows the e-Commerce growth in the last year.

Table I
Growth in Online Retail Sales in US and Europe; Values expressed in billions (source: Forrester Research)

	On-line Retail 2010	On-line Retail 2011	Growth
US	258 USD	287 USD	11%
West Europe	133 EUR	150 EUR	13%
UK	46.2 EUR	50.9 EUR	10%
Germany	30.9 EUR	34.1 EUR	10%
France	17.9 EUR	20.1 EUR	12%
Italy	7.4 EUR	8.6 EUR	16%

For an e-Commerce system the ability to manage product bundling generation can be particularly useful in satisfying consumer needs and preferences and thus the question of finding an optimal bundle configuration becomes crucial. Looking for an optimal bundling means looking for a partitioning of items into bundles that promotes products with some features, best fits the retailer's needs and maximizes product compliance within the bundle. According to the current literature, the bundling problem is a challenging task, whose applications go further the desktop and web applications. This problem has been recently modeled as a constraint-based problem [1], [2], where the bundle represents a set of n variables which are the candidate products for the bundle. Consequently, the aim is to find an assignment to all variables in the model that does not violate any hard constraint and optimizes the bundles satisfying soft constraints.

In this paper, the problem of finding an optimal product bundle consists in finding a valid product bundle made of items that (i) minimize dead stock by proposing products with the highest availability, (ii) maximize revenue for the merchant by preferring the most profitable products, and (iii) ensure the compliance with a set of constraints by choosing those products which satisfy better the elicited requirements.

The remainder of this paper is organized as follows: Section 2 introduces the bundle problem and the related works, Section 3 introduces the state-of-the-art algorithms for mining association rules, Section 4 describes the proposed system for generating and suggesting product bundle, Section 5 shows the proposed engine, Section 6 provides experimental results, and Section 7 outlines conclusions and future directions.

II. Product bundling problem

Bundling problem can be classified as a problem of *product bundling*, which is defined as "the integration and sale of two or more separate products or services at any price", and *price bundling*, which is the "sale of two or more separate products in a package at a discount, without any integration of the products" [3]. The main difference between these two concepts is that price bundling does "not create added value to customers and thus a discount must be offered to motivate at least some customers to buy the bundle" [3].

In our work, we consider the product bundling problem, searching for a package which satisfies a set of constraints specified by merchants or by customers. As the product bundle aims to quickly sell to customer a set of products, the problem solution relies on finding a bundle that maximizes the compatibility of the products within the bundle, and satisfies at the same time customer preferences and merchant requirements.

Bundle generation involves most aspects of human information processing. Indeed, the user is asked for visually inspect the interesting products, reading and comprehending the product details in order to find a good set of products which satisfy his demand. The optimal bundle is expected to better support the user in this task by providing directly a bundle of interesting products that meets his demand.

Various models for predicting the selected items within bundle have been proposed. The study of bundling in the economics literature was started by Palfrey [4] and extended by Chakraborty [5]. Product bundle can be suggested to users mainly by means of two approaches: explicit or implicit.

The explicit approach consists of a bundle creation which is directly managed by the retailer who controls the operation of product matching as he wishes. This approach is followed by some e-Commerce platforms like *Magento*, which is able to combine items together.

Instead, the implicit approach is based on an automated bundle generation which is totally managed by the system. In this sense *Amazon.com* suggest as a bundle a set of products that are often booked together by means of data mining techniques. Some implicit strategies [1], [2], [6] have been proposed in literature. Zanker et al. [1], [2] proposed a constraint-based approach for the product bundling problem. They defined the problem as a Constraint Satisfaction Problem (CSP). According to Tsang [7], Constraint Satisfaction Problem (CSP) consist of a set of variables with finite domains and a set of constraints that describe allowed value assignments for variables. This problem is formulated as a triple (X, D, C) where Z is a set of variables $(X = \{x_1, x_2, \ldots, x_n\})$, D is a function that maps each variables in Z to a domain $(D = \{d_1, d_2, \ldots, d_n\})$, and C is a set of constraints $(C = \{c_1, c_2, \ldots, c_m\})$.

In Zanker approach, the domain model is represented by a tuple that consists of the set of variables subdivided into several disjoint subsets, modeling the user model, the system context, the product classes and the product properties; the sets of corresponding domains for the product classes and their properties; the set of constraints subdivided into hard and soft. Soft constraints may be violated by variable assignments, but each violation is typically associated with a penalty value. In order to reach an optimal solution, the sum of penalty values of violated constraints has to be minimized. Once the CSP model is generated, the authors [1], [2] invoked a JAVA solver in order to find an assignment to all variables introduced that does not violate any hard constraint and minimizes the resources.

The remaining approaches presented in the literature [8], [9] assimilated product bundling problem to a recommendation task where the bundle is the set of recommended items without taking compliance issues into account.

In previous works, we successfully tested Genetic Algorithms (GA) [10]–[12] by re-arranging the disposition of different web interfaces as Web form fields and Web pages with images, text areas and buttons on different pages according to some constraints. These research papers are focused on automatically arranging elements in unforeseen layouts by means of GA.

Moreover, in a previous paper [13], we proposed genetic algorithms as an effective means for finding a product bundle that satisfies user preferences and product constraints (named soft and hard constraints according to their priority). The previous paper considered an artificial catalog made of 507 products with a set of constraints which are introduced in order to test the effectiveness of the genetic engine. We considered two kinds of constraints: *soft constraints*, regarding preferences about products matching, and *hard constraints* regarding some particular constraints commonly considered fundamental as the compatibility of different products (the properties of products are compared in order to choose appropriate and compatible products within the bundle). Soft constraints had to be satisfied in order to increase the fitness value, while hard constraints were verified in order to make solution valid. Bundle of items which did not satisfy hard constraints was an illegal solution. In the proposed approach we design a system which is able to elicit implicit constraints avoiding the pure explicit approach to constraint definition and is able to search for an optimal bundle taking into

account: (i) *compliance*, as the maximization of compliance of products within the bundle, (ii) *merchant requirements*, as a set of merchant requirements or best practices in building the bundle, and (iii) *preferences*, as a wish list made explicit or implicit by the end user.

III. Mining association rules

Let $I = \{i_1, i_2, \ldots, i_N\}$ be a set of N distinct items or products. A transaction is a record of one or more items collected from a finite item domain, and a dataset D is a collection of transactions. In general, a set of items is called an *itemset*. Each itemset has an associated statistical measure called support [14], defined as the number of itemset occurences in the dataset. In other words, for an itemset $A \subset I$, $supp(A) = s$, if the fraction of transactions in D containing A equals s. Itemsets whose support is higher than a given threshold are defined as *frequent*.

A primer algorithm to discover frequent itemsets in a database is the *Apriori* algorithm proposed by Agrawal and Srikant [15]. This algorithm is level-wise, as it considers itemsets with different cardinality at each step. At step k frequent itemsets having k items are available in F_k. Although the Apriori algorithm is able to prune large parts of the search space, it can be still computationally expensive on large databases, and several improvements to the way frequent itemsets are processed and stored (e.g. see reference [16]) has been proposed. Han et al. [17] came up with a solution based on compact tree structure, named FP-Tree, on which to apply a partition-based divide and conquer mining strategy. This approach has proved to perform faster than other techniques.

An association rule is an implication of the form $A \Rightarrow B$, where $A, B \subset I$, and $A \cap B = \emptyset$. A is called the antecedent of the rule, and B is called the consequent of the rule. It has a measure of its strength called *confidence* defined as the ratio, $supp\,(A \cup B)/supp\,(A)$ where $A \cup B$ means that both A and B are present.

An example of such an association rule is the statement that 80% of transaction that purchase bread and butter also purchase milk. The antecedent of this rule consists of bread and butter and the consequent consists of milk. The confidence factor of this rule is 80%. The problem of mining association rules between sets of items in a large database of customer transaction consists in generating all the association rules that have a user-specified minimum support and a user-specified minimum confidence.

In this paper, we adopt the association rules extracted by the transactional database in order to discover products that can be proposed in a same product bundle.

IV. Intelligent Bundle Suggestion and Generation System

Our work is aimed at developing an Intelligent Bundle Suggestion and Generation (IBSAG) system able to support the merchant to generate product bundle and able to provide offers that match a specific user's taste. The proposed system, depicted in Fig.1, is split in four steps: (i) Definition of merchant constraints, (ii) Acquisition of implicit constraints, (iii) Bundle Generator, (iv) Bundle Suggestion.

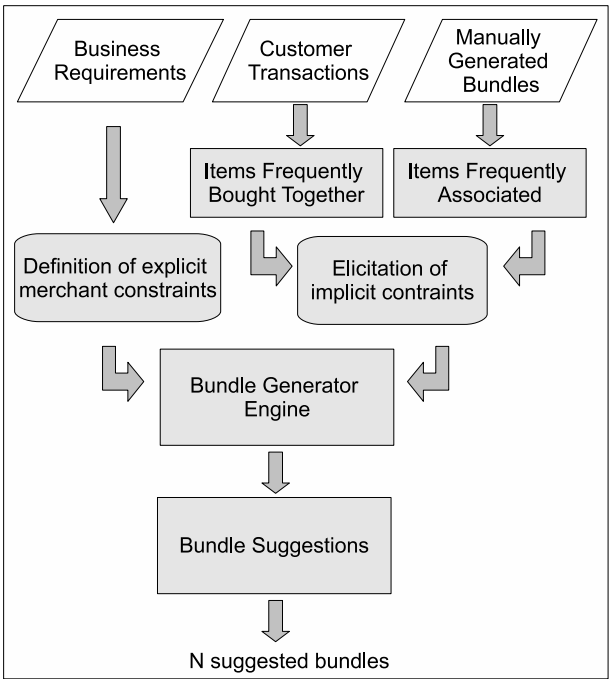


Figure 1. Overview of the construction and operation of IBSAG system.

The first step is the definition of merchant requirement about the bundle, e.g., he defined as preferred bundle, the bundle that promotes products with high dead stock. The second step consists in logging transaction data when users acquire a set of products and when users compose a desired bundle. Transaction logs are used as starting point to extract the support by means of A-priori algorithm. Furthermore, the generator engine seeks a solution able to satisfy the constraints, and finally a suggestion system propose to the target customer the bundle he/she could prefer.

In other words, the customer will receive a suggested bundle which, on the one hand, satisfies merchant constraints guaranteeing a solution made by combinable items and, on the other hand, selects the bundle with the highest score of preference index. Preference index P_b of the bundle b is evaluated per customer by adding the explicit (where available) or implicit rating assigned to each item within the bundle, as expressed in (1):

$$P_b(z) = \frac{1}{N_b} \cdot \sum_{i=1}^{N_b} r_i(z) \tag{1}$$

where $r_i(z)$ is the rating given to item i by the customer z, and N_b is the number of items within the bundle b.

V. Bundle Generation Engine

Bundle Generation Engine is aimed at finding a solution that satisfies a set of requirements such as retailer needs and the compliance within the bundle because some products could be not compatible with other ones.

The engine is based on a Genetic Algorithm [18], whose structure is outlined in Fig.2.

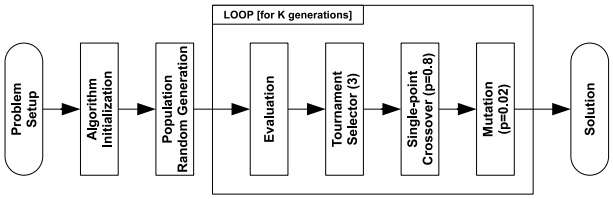


Figure 2. Algorithm structure.

The initial population is randomly built. For selection, we adopt a tournament operator in order to reduce the effect of fitness scaling, since each individual is evaluated according to a fitness function defined on logical basis, as described below. Crossover and mutation are implemented using a bid strategy, i.e. attempting to make a valid solution within a given number of trials. Elitism replaces random individuals with best individuals in order to improve performance, as this strategy does not require the sorting of the population before application.

A. Chromosome structure

We adopt a chromosome composed of different objects, where every gene represents a particular product belonging to the merchant's catalogue. The phenotype of a chromosome is a particular product bundle, indeed grouping different instances of these classes we obtain different individuals. Furthermore, every gene can assume values within a defined allele set. Chromosome structure is depicted by Fig.3 where each gene represents a product whose reference reserves 64 bits.

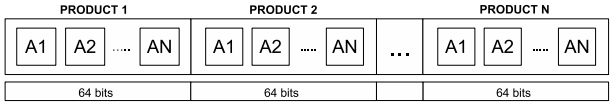


Figure 3. Chromosome structure.

B. Fitness Function

The fitness function of an individual x is defined as a convex combination which models the trade-off between retailer needs and compliance with constraints. In particular, it is a function (to maximize) assuming values in the unit interval $[0, 1]$ and defined as:

$$fitness(x) = \sigma \cdot C(x) + (1 - \sigma) \cdot M(x) \qquad (2)$$

where $\sigma \in [0, 1]$, $M(x)$ takes into account retailer needs, and $C(x)$ is the degree of compliance with constraints. The fitness function ranges from $[0, 1]$, 0 being the worst result and 1 the best, i.e. the bundle satisfies all the constraints by choosing the products which have the highest compliance with retail requirements. Several constraints are conflicting so that fitness equal to 1 is an ideal solution that is never reached.

Among retailer needs we implemented two kind of requirements: suggesting products with high dead stock (see Eq.4) and selling products which can maximize net profit (see Eq.5). So that, $M(x)$ consists of two factors $G(x)$ and $U(x)$, so that Eq.2 is updated as follows:

$$fitness(x) = \sigma \cdot C(x) + (1 - \sigma) \cdot (G(x) + U(x)) \qquad (3)$$

In Eq.4, retailer dead stock is considered:

$$G(x) = \frac{1}{N} \cdot \sum_{i=1}^{N} \frac{g_i(x)}{\max(g_i)} \qquad (4)$$

where N is the number of products in the chromosome and $g_i(x)$ is the availability of the i-th product within the retailer shop.

While, according to Eq.5 the net profit factor is evaluated as:

$$U(x) = \frac{1}{N} \sum_{i=1}^{N} \frac{p_i(x) - c_i(x)}{\max(p_i - c_i)} \qquad (5)$$

where N is the number of products in the chromosome, $p_i(x)$ and $c_i(x)$ are respectively the revenue and cost of the i-th product within the retailer shop.

Instead the degree of compliance with constraints $C(x)$ is defined as the weighted mean of different constraints c_k. When verifying a couple of n-tuples, there exists the related associative rule, the solution is increased by a score equal to the confidence of the specific rule. The more associative rules with a confidence equal to 1 there exist, the more $C(x)$ tends to 1 (upper limit never reached).

VI. Experimental Results

A. Experimental setup

In our experimentation we consider Contoso BI Demo Dataset and a transactional database of a real e-Commerce platform of Poste Italiane. Contoso BI Demo Dataset is a fictitious retail demo dataset defined by Microsoft and used for presenting Microsoft Business Intelligence products. It consists of about 19,000 customers, 2,517 products and 12,627,608 sold items. The Poste Italiane dataset[1] refers to the e-Commerce orders from January 1st, 2008 to December 31st, 2011 and consists of 1,483 products and 6,820 customers with 9,462 sold items. The data about dead stocks

[1]Poste Italiane sells different products by means of an e-Commerce platform which includes a wide set of products in different categories such as Home, Furniture, Photography, Books & magazines, Mobile phones & communication, Office equipment, containers, stamps and postal items.

and profits have been randomly simulated with values in [10,50] and [10%,20%] interval respectively.

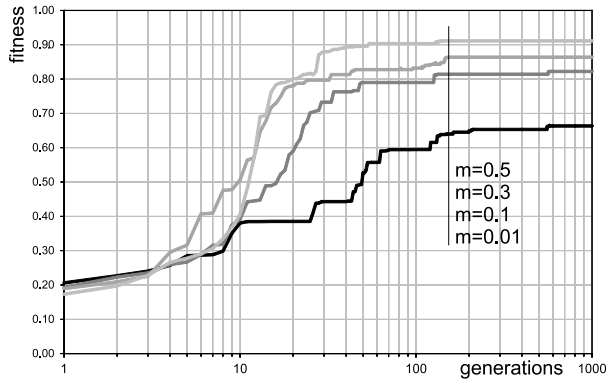


Figure 4. Average best individual fitness of 10 different runs at varying mutation rate.

B. Results

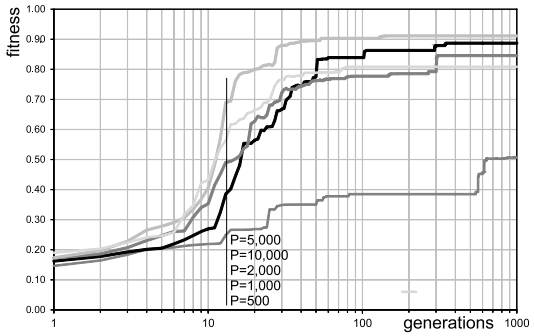
In our bundling problem we look for the optimal 4-item bundle, on the one hand, able to satisfy merchant requirements, promoting products with high dead stock values and preferring products that maximizes net income, on the other hand, able to include within the bundle products which are correlated each other by means of some extracted associative rules.

Market basket analysis is performed by means of FP-growth algorithm with a minimum support of 2.0e-04 and 1.0e-04 when Poste Italiane dataset and Contoso dataset are respectively considered.

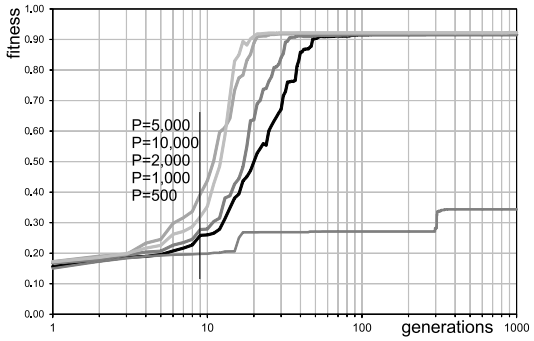
For the experimentation, the chosen algorithm parameters are: (i) crossover rate equal to 0.8, (ii) selection tournament equal to 3, (iii) generation limit equal to 1000 (no significant improvement after the 1000th generation), (iv) elitism equal to 3 and (v) σ equal to 0.8 (see Eq.2).

As first step, we investigated the problem convergence at varying mutation rate (i.e. 0.01, 0.1, 0.3, 0.5). As depicted in Fig.4, repeating 10 different runs for different problem configurations, we observe the benefit of an high mutation rate by plotting the fitness average of best individuals.

Moreover, we prove the effect of population size, observing how the fitness converges at different population size (i.e. 500, 1000, 2000, 5000 and 10000 individuals) with a randomization rate equal to 0.9 and a mutation rate of 0.5 (see Fig.5). As expected, population size has an impact on discovery of optimal solutions and best results occur when population size is 5000 or 10000. Instead adopting few individuals the algorithm is not able to converge when Contoso dataset is taken into account and reaches low fitness solution when Poste Italiane dataset is considered. The high value of needed individuals is justified by the wide search space due to the problem configuration who has to find the



(a) Poste dataset



(b) Contoso dataset

Figure 5. Average best individual fitness behavior by varying the population size.

four items within a catalog made by 1,483 (Poste Italiane dataset) and 2,517 products (Contoso dataset).

		Product 1	Product 2	Product 3	Product 4
Bundle1 [fitness = 0.13]	ID	LE420000	02EB0000	MC060000	01YO0000
	Description	Samsung TVC LCD 42	Batteria di pentole acciaio inox 18 10	La moneta Italia campione del	Collier Zodiaco Pesci
	Net Profit	€ 139.79	€ 33.80	€ 3.90	€ 5.60
	Price	€ 699.00	€ 169.00	€ 19.50	€ 28.00
	Dead stock	50	50	50	50
Bundle2 [fitness = 0.77]	ID	TE090000	LI020000	TE100000	TE080000
	Description	Teletubbies - Guarda la'!	Tutto il Grillo che conta. Monologhi	Teletubbies - ninna nanna	Teletubbies - animali grandi...
	Net Profit	€ 1.98	€ 1.10	€ 1.48	€ 1.98
	Price	€ 9.90	€ 11.05	€ 9.90	€ 9.90
	Dead stock	50	10	30	50
Bundle3 [fitness = 0.90]	ID	TE090000	TE100000	TE080000	99zf0000
	Description	Teletubbies - Guarda la'!	Teletubbies - ninna nanna	Teletubbies - animali grandi...	LED UE32B6000VW 32 100HZ
	Net Profit	€ 1.98	€ 1.48	€ 1.98	€ 149.80
	Price	€ 9.90	€ 9.90	€ 9.90	€ 749.00
	Dead stock	50	30	50	50

Figure 6. Example of product bundles with different fitness values.

Fig.6 summarizes a qualitative analysis of the results obtained during a run of the algorithm on the Poste dataset, three specific bundles, which are obtained during an evolutionary process, are depicted. The presented bundles show the phenotype of the best individual respectively after 5, 50, and 1000 generations. The first bundle, obtained after 5 generations, has a very low fitness (0.13), and this because the contribution given by associative rules is poor (there exist just two rules involving one single product of the considered bundle), though fully respecting the dead-stock

and profit requirements. Viceversa, the second and the third bundles show a better fitness, even if they have lower dead stocks and profits; in this case the satisfied associative rules are 12 and involve 3 products of the bundle (e.g., $TE090000, TE100000 \Rightarrow TE080000$, where $TE090000$, $TE100000$, $TE080000$ are three product IDs). We can conclude that fitness can measure the quality of provided solutions and the proposed algorithm converges toward an optimal solution.

VII. Conclusions and future work

While existing e-Commerce frameworks such as Magento and ATG Commerce implemented bundling functionalities which are completely chosen by the merchants who specify the items in the bundle, we proposed a generative solution for the product bundling problem, keeping into consideration merchant requirements and product compliance within the bundle. The resulting solution can be used as a robust starting point aimed to be refined by software engineers. Experimentation provided very encouraging results, proving the ability of the proposed algorithm in converging towards solutions with high fitness, also in presence of different constraints. However, we aim to investigate in the future different evolutionary techniques, e.g., making the genetic algorithm interactive poses additional interesting questions to be answered about how to sample the search space and to gather user's feedback.

Acknowledgments

This work was partially supported by MSE under the Intelligent Virtual Mall (InViMall) Project MI01-00123.

References

[1] M. Zanker, D. Jannach, M. Silaghi, and G. Friedrich, "A distributed generative CSP framework for multi-site product configuration," in *Cooperative Information Agents XII*, ser. Lecture Notes in Computer Science, M. Klusch, M. Pechoucek, and A. Polleres, Eds. Springer Berlin / Heidelberg, 2008, vol. 5180, pp. 131–146.

[2] M. Zanker, M. Jessenitschnig, and W. Schmid, "Preference reasoning with soft constraints in constraint-based recommender systems," *Constraints*, vol. 15, pp. 574–595, 2010.

[3] S. Stremersch and G. J. Tellis, "Strategic Bundling of Products and Prices: A New Synthesis for Marketing," *Journal of Marketing*, vol. 66, no. 1, pp. 55–72, Jan/00 2002.

[4] T. R. Palfrey, "Bundling decisions by a multiproduct monopolist with incomplete information," *Econometrica*, vol. 51, no. 2, pp. 463–83, 1983.

[5] I. Chakraborty, "Bundling decisions for selling multiple objects," *Economic Theory*, vol. 13, no. 3, pp. 723–733, 1999.

[6] T. Kowatsch, W. Maass, A. Filler, and S. Janzen, "Knowledge-based bundling of smart products on a mobile recomendation agent," in *Proc. of 7th Int. Conf. on Mobile Business, 2008. ICMB '08.*, july 2008, pp. 181 –190.

[7] E. Tsang, *Foundations of Constraint Satisfaction*. London, UK: Academic Press, 1993.

[8] L. Guo-rong and Z. Xi-zheng, "Collaborative filtering based recommendation system for product bundling," in *Proc. of Int. Conf. on Management Science and Engineering, 2006. ICMSE '06.*, oct. 2006, pp. 251 –254.

[9] S. Sedeh, M. Nematbakhsh, and F. Mofakham, "A bidding strategy in combinatorial auctions," in *Proc. of 2nd Int. Conf. on Intelligent Systems, Modelling and Simulation. ISMS '11.*, jan. 2011, pp. 422 –427.

[10] L. Troiano, G. Cirillo, R. Armenise, and C. Birtolo, "A preliminary experience in optimizing the layout of web pages by genetic algorithms to fit mobile devices," in *Intelligent Systems Design and Applications, 2009. ISDA '09. 9th Int. Conf. on*, 30 2009-dec. 2 2009, pp. 1055 –1061.

[11] L. Troiano, C. Birtolo, and M. Miranda, "Adapting palettes to color vision deficiencies by genetic algorithm," in *GECCO '08: Proceedings of the 10th annual conference on Genetic and evolutionary computation*. New York, NY, USA: ACM, 2008, pp. 1065–1072.

[12] L. Troiano, C. Birtolo, R. Armenise, and G. Cirillo, "Web form page in mobile devices - Optimization of layout with a simple genetic algorithm," in *Enterprise Information Systems 2009. ICEIS '09, 11th Int. Conf. on*, J. Cordeiro and J. Filipe, Eds., 2009, pp. 118–123.

[13] C. Birtolo, D. De Chiara, M. Ascione, and R. Armenise, "A generative approach to product bundling in the e-Commerce domain," in *Nature and Biologically Inspired Computing (NaBIC), 2011 Third World Congress on*, oct. 2011, pp. 169 –175.

[14] R. Agrawal, T. Imieliński, and A. Swami, "Mining association rules between sets of items in large databases," in *SIGMOD '93: Proc. of the 1993 ACM SIGMOD Int. Conf. on Management of data*. New York, NY, USA: ACM, 1993, pp. 207–216.

[15] R. Agrawal and R. Srikant, "Fast algorithms for mining association rules in large databases," in *VLDB'94, Proc. of 20th Int. Conf. on Very Large Data Bases, September 12-15, 1994, Santiago de Chile, Chile*, J. B. Bocca, M. Jarke, and C. Zaniolo, Eds. Morgan Kaufmann, 1994, pp. 487–499.

[16] J. Janas, "An enhanced a priori algorithm for mining multidimensional association rules," in *Information Technology Interfaces, 2003. ITI 2003. Proc. of the 25th Int. Conf. on*, June 2003, pp. 193–198.

[17] J. Han, J. Pei, Y. Yin, and R. Mao, "Mining frequent patterns without candidate generation: A frequent-pattern tree approach," *Data Min. Knowl. Discov.*, vol. 8, no. 1, pp. 53–87, 2004.

[18] D. E. Goldberg, *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison Wesley, 1989.