From Real Purchase to Realistic Populations of Simulated Customers

Philippe Mathieu, Sébastien Picault

From Real Purchase to Realistic Populations of Simulated Customers

Philippe Mathieu and Sébastien Picault

Université Lille 1, Computer Science Dept. LIFL (UMR CNRS 8022),
Cité Scientifique 59655 Villeneuve d'Ascq Cedex, France
firstname.surname@univ-lille1.fr
http://www.lifl.fr/SMAC/

Abstract. The use of multiagent-based simulations in marketing is quite recent, but is growing quickly as the result of the ability of such modeling methods to provide not only forecasts, but also a deep understanding of complex interactions that account for purchase decisions. However, the confidence in simulation predictions and explanations is also tightly dependent on the ability of the model to integrate statistical knowledge and the situatedness of a retail store. In this paper, we propose a method for automatically retrieving prototypes of consumer behaviors from statistical measures based on real data (receipts). After preliminary experiments to validate this data mining process, we show how to populate a multiagent simulation with realistic agents, by initializing some of their goals with those prototypes. Endowed with the same overall behavior, validated in earlier experiments, those agents are put into a spatially realistic store. During the simulation, their actual actions reflect the diversity of real customers, and finally their purchase reproduce the original clusters. Besides, we explain how such statistically realistic simulation may be used to support decision in retail, and be extended to other application domains.

Keywords: Agent-based Simulations, Knowledge discovery, Marketing, Interactions

1 Introduction

Since several years, individual-based models and multiagent-based simulations have been used to enhance the understanding of complex marketing issues and support decision in the context of retail stores [1, 2]. Due to the introduction of fine-grained information regarding individual behaviors, agent-based models are able to provide not only global predictions (such as the quantities of transactions or revenue) but also insights concerning the reasons that make marketing techniques efficient or not.

Classical marketing analysis techniques, used e.g. to segment customers into subgroups with similar needs, or to detect items that are frequently bought together, consist in retrieving global information from large databases through data mining algorithms [3, 4]. For instance, association rules capture co-occurrences between items in customer baskets, and thus allow the retailer to offer promotions on frequently associated items, or to propose relevant similar products. But, since those techniques only capture statistical features of customer behavior, without being able to provide any kind of causal

explanation, their quality is highly dependent on the application that uses the retrieved knowledge [5]. Besides, the data collected in retail stores result from a complex decision process, which is affected by seasonal, geographical, cultural, environmental factors, by demographic and psychographic variations of the customers, by the brand management, and by in-store events such as promotions. Thus it is difficult to assess the stability of the rules that are built from the data, and quite impossible to predict how changes in the store management affect the rules.

On the contrary, agent-based models allow to take into account individual preferences and even psychological expertise [6], so as to build an accurate description of the motivations and needs of each customer. Then it is the actions of those simulated customers that are responsible for the purchases that are predicted. Hypotheses regarding the factors that influence sales are made explicit in the model: they can be understood and examined by experts, and validated (or not) through an appropriate experimental setup.

As a counterpart, individual-based models, especially when they involve cognitive agents (e.g. for accounting psychological motivations [6]), tend to require too much expertise, which is not always easy to acquire or implement. In addition, few store simulation models do take into account the spatial issues which are considered crucial in retail stores, such as shelves allocation, items placement, checkout sizing, point of sale display, etc. In a previous work, we addressed those questions in order to build a simulation of supermarkets, where the agents were situated in a realistic environment [7]. *This situatedness is necessary for raising multiagent systems from casual, ad hoc simulations, to full decision support systems*, able to predict how the clients react to changes in the spatial organization of the store, marketing events (e.g. discount, publicity...), and competitor shops.

In this paper, we propose a simulation approach which does not rely much on expert knowledge; instead, it tries to retrieve as much information as possible from retail data (e.g. receipts) as in classical basket analysis methods; but, this knowledge discovery process is used to initialize the purchase preferences of the population of simulated customers with statistically realistic traits. Combined with a model of customer behavior, those traits produce statistically realistic purchase when the agents are acting in their environment.

Since we designed and validated a model of customer behavior in a previous work [7], we will focus in this paper on the realism of customer populations, i.e. on a purchase decision model which can mimic actual behaviors.

The main classical technique for information retrieval in actual data is the affinity analysis [3, 8], which is based on the census of item co-occurrences in the purchases. It can be directly applied to actual receipts. On this basis, association rules between items (i.e. $X \rightarrow Y$ where X, Y are disjoint sets of items) can be inferred [3], together with a *support* (proportion of purchases which include both X and Y) and a *confidence* (conditional probability of buying the articles of Y when those of X are in the basket).

This approach is very helpful for cross-selling or up-selling, and to some extent it can provide indications in product placement (e.g. try to associate in the shelves items that are frequently bought together). Its first limitation is the computational time, which grows at least with the cube of the number of items [9]. But also, it is quite difficult

to use association rules to drive the purchase decisions of agents, since a rule can only suggest items that are *related to others*, but not how to bootstrap the basket.

An alternative approach consists in trying to predict *shopping lists* from the existing receipts. For instance in [10] this method is implemented in a personal assistant that learns individual purchase habits, so as to remind the customers of their most probable needs during the shopping trip, and to propose personalized promotions. The purpose of this application is very far from ours, and the classification methods that are used do not build any symbolic description of the shopping list, but only a prediction over rough product categories. Nevertheless, this work assesses the possibility to perform some kind of induction over real receipts so as to identify an underlying shopping list.

In our proposal, we try to join the identification of similar customers (like in classical market segmentation) and the induction of abstract descriptors for those clusters, specifically prototypical shopping lists. Then we use the association of clusters and shopping lists to generate profiles of agents which buy close items. Hence the approach we propose follows a kind of methological loop. We define a representation frame for transactions (e.g. receipts) and their abstract description: prototypes built by generalization. Afterwards those prototypes are used to initialize the agents in the simulation and the simulated transactions can be in turn analysed.

The paper is organized as follows: section 2 presents briefly the context of the grocery store simulation that has been designed and tested in [7, 11]; especially we explain how shopping lists are used to induce purchase preferences. Section 3 describes the way we represent relevant information to identify and characterize items, transactions and prototypes. Section 4 presents the data mining process that builds prototypes from transactions, and section 5 how the overall procedure has been validated. Finally we explain in section 6 how to use our approach within multiagent simulations.

2 Context of the simulations

The knowledge discovery process we propose in this paper has been tested within an existing grocery store simulation [7, 11], endowed with a realistic environment and populated with behaviorally convincing artificial customers.

2.1 An Interaction-oriented model

In this work, we had to acquire expertise and build the simulation model in a quite incremental and empirical way. Thus the modeling method had to be highly understandable by experts outside the field of computer science, e.g. psychologists or marketing advisors, and enable a step-by-step design of behaviors. Therefore, we used the principles of the 'Interaction-Oriented' approach [12]. In this method, each relevant entity of the model is represented by an agent, without any prior distinction between "true" agents and resources or objects; each behavior is modeled by a separated piece of code called an "interaction", i.e. a sequence of actions between a source agent and one or more target agents, controled by some conditions. The interaction is realizable if the source and target agents fulfil the conditions.

Agents and interactions can be developed in independent libraries, then the simulation is designed by assigning interactions to pairs of agents (in an 'interaction matrix' – see table 1). It is a visual way to express what families of agents are allowed to interact, and with what interactions. This interaction matrix is processed by a generic simulation engine, which essentially evaluates what interactions can be realized, between what agents, and subsequently determines the actions to be performed by each agent.

Actually, describing the action capabilities of the entities of the model in terms of interactions that can be performed or undergone, instead of behaviors embedded in the agents, is very close to the theory of affordances [13]. Thus, it makes this quite natural to use for psychologists and facilitates knowledge acquisition.

2.2 Model of a customer agent

Table 1. The interaction matrix that defines the behavior of all agents in our simulation. For instance, the intersection between line 'Customer' and column 'Item' contains two interactions, 'Take' and 'MoveTowards', which means that a customer agent may either take an item agent, or move close to it, depending on the priority (first number – here, taking an item has the highest) and on the distance between agents (second number), assuming that the conditions of the interactions are fulfiled for both agents. The $\emptyset$ column contains reflexive interactions (i.e. where the target agent is the source itself). Empty columns and lines have been removed.

Source \ Target	$\emptyset$	Customer	Item	Checkout	Queue	Door
Customer	Wander (0) GoToPlace (1, ∞)		MoveTowards (2, 10) Take (4, 1)	MoveTowards (3, 10)	StepIn (5, 2) MoveOn (6, 1) WalkOut (7, 1)	Exit (8, 1)
Item		Notify (1, 10)				
Sign		Notify (1, 10)				
Checkout	Open (10) Close (10)	Notify (1, 15) CheckOut(7, 1)			Handle (8, 1) ShutDown (9, 1)	
Door	SpawnCustomer(1)	Notify(1, 10)				

In the work presented here, we use a model of customer behavior that was previously developed for the simulation of a retail store, aimed at studying human vendors confronted to artificial clients [11]. Thus, the overall behaviors of simulated customers have been validated by marketing experts and are set once and for all in the work described here. In what follows, the vendor was removed for studying only the clients. The corresponding interaction matrix is shown on table 1.

To summarize, it is assumed here that all clients have the same overall behavior, but *differ in their needs*: thus they are endowed at startup with a *shopping list*, which specifies more or less precisely what items they are likely to buy in the store. The needs may be specified with accuracy, e.g. "SodaCola light, family pack", or with vague indications, e.g. "spring water", which can match much more actual items. The purchase decision is implemented by the 'Take' interaction, which consists of two conditions (the target agent matches an item of the shopping list of the source agent; and: the source has enough money) and one action (put the target in the basket of the source).

During the simulation, interactions occur according to the interaction matrix and the state, perceptions and positions of agents, producing a consistent consumer behavior: artificial clients try to find all items which figure in their list, within a limited amount of time. They may be endowed with a mental map of the shop or with no prior knowledge; they also are notified by panels, checkouts, items, etc. about relevant information to help them.

In the original work, the shopping lists were either computed through a pure random process (following a Poisson distribution based on the average basket size), or implemented "by hand", according to a specific mise en scène, in a deliberate intend to put the vendor in a problematic case. In the experiments described below (see section 5), those lists have been built through the knowledge retrieval algorithm we propose.

3 Knowledge representation

Before describing the data mining process which builds prototypes from transactions, we must explain how we represent both kinds of information. This step may require the intervention of a marketing expert, but afterwards the knowledge retrieval process is automatic.

3.1 Items identifiers: from SKUs to meaningful descriptions

In retail stores, each unique product is usually identified by a "Stock-Keeping Unit" (SKU) in order to track availability and demands. The SKU does not necessarily carry any special meaning regarding the nature and characteristics of the product. Other methods, such as the Universal Product Code (UPC), European Article Number (EAN), etc. can be used as well. It may also happen that the actual purchase are anonymised through an automatically generated identifier, e.g. in order to perform basket analysis under strong confidentiality constraints.

Those identification methods are actually not well suited for extracting more than co-occurence rules. Relevant marketing knowledge (e.g. product family, quality, relative price, brand image, organic label...) must be added to characterize the products so as to allow an explanatory analysis.

In our approach, each unique product is identified by a **tuple of strictly positive integers**, which encodes the features values that are considered relevant in the application context. For instance, if the relevant features are the brand, the product family and the details (e.g. respectively "SodaCola", "beverage", "soda with cola"), then products will be identified only by a triple of integers, e.g. `(31, 4, 15)`. This allows a representation of all products at an arbitrary fine level, including specific labels such as "organic", "fair trade" or "gluten-free". Also, continuous values such as the price or weight may be encoded through a prior categorization (e.g. 1 for "cheap", 2 for "average", 3 for "expensive"; or 1 to 4 for small to extra large packings). To some extent, the mapping between SKU (or other identification systems) to this kind of integer tuple can be performed automatically, through an appropriate join of databases. Yet, the selection of features that have to be taken into account for providing a relevant description of the products may involve a marketing expertise.

3.2 Transactions and prototypes

Our data mining process relies upon the recording of actual purchases. The simplest way to do this is to retrieve information directly from the receipts. A *transaction* can be computed as a mere enumeration of the unique products of a receipt. Quantities are not taken into account as such (exactly like in the classical affinity analysis process), though they could be added as a trait of each item (like other continuous features, see above). Thus, from a list of SKU, we build a set of integer tuples.

From those transactions, the knowledge retrieval process consists in building *prototypes*, which are aimed at describing an "abstract receipt" so as to characterize clusters of receipts. In order to do so, we introduce the concept of *prototype item*, which are also integer tuples, but **allowing the value 0 as a wildcard**. For instance, a product characterized as "any brand", "beverage", "soda with cola", could be described by the triple `(0, 4, 15)`. The null tuple `(0, 0, 0)` means "any item". A *prototype* is simply a set of prototype items.

Those prototypes, built from real data, may also be used as a "shopping list" for simulated customers, because it may often happen that only few traits of the desired items are specified. For instance, Mr Smith always buys "soda with cola" but is indifferent to the brand, while Mr Wesson is likely to buy any organic yoghurts from the brand "Yoopla". The use of the 0 wildcard is very helpful for expressing such vague wishes.

In the next section, we show how such prototypes are actually built from the transactions.

4 Steps of the data mining process

In order to analyse the purchases, we proceed as follows:

1. The transactions database is partitioned into clusters (this requires first to define a distance measure between receipts, which is itself based on a distance measure between items).
2. For each cluster:
   (a) all items that appear in the transactions are in turn classified so as to build prototype items ;
   (b) the prototype composed of the union of prototype items is scored against the transactions of the cluster.

4.1 A measure of item similarity

Since some items of a customer's shopping list may be not fully specified, it is expected that several customers who have the same prototype (i.e. the same shopping list) will not get exactly the same items. Thus, if the distance between transactions relies only upon the equality of items, it is likely to produce a defective clustering. Instead, we propose to modulate the comparison between transactions, by taking into account the distance between items.

A simple way to do so is to compute a Hamming-like distance (or conversely, a Hamming-like similarity index). If two items are encoded by the tuples $I = (f_1, ..., f_n)$ and $I' = (f'_1, ..., f'_n)$, the similarity between items is defined as: $\sigma(I, I') = \frac{1}{n} \sum_{i=1}^{n} \delta(f_i, f'_i)$ where $\delta(f_i, f'_i) = 1$ if $f_i = f'_i$ or $f_i = 0$ or $f'_i = 0$, and 0 otherwise (note that this definition works fine for actual items and for prototype items as well).

4.2 A measure of transaction similarity

In order to compare transactions, we started with a well-known measure which is frequently used for measuring similarities between sets: the Jaccard index [14]. It is defined for any subset X, Y as follows: $J(X,Y) = \frac{|X \cap Y|}{|X \cup Y|} = \frac{|X \cap Y|}{|X|+|Y|-|X \cap Y|}$

As it has been previously said, the rough use of the Jaccard index cannot fit our needs, since it would make no difference between disjoint sets and sets that contain very similar but different items. Thus we propose an extension of the Jaccard index, based on the distance between items. It consists in computing the best matching score between items of both transactions (thus we called it the *best-match Jaccard index*, denoted by J_{BM}). To compute it between a transaction $T = \{I_1, ..., I_p\}$ and another transaction $T' = \{I'_1, ..., I'_q\}$, we follow these steps:

1. Compute the matching matrix $(\sigma_{i,j})$ with $\sigma_{i,j} = \sigma(I_i, I'_j)$
2. For each k between 1 and $\min(p,q)$:
   (a) compute $\mu_k = \max_{i,j}(\sigma_{i,j}) = \sigma_{i^\star,j^\star}$ (if several (i,j) values verify $\mu_k = \sigma_{i,j}$, we take one pair $(i^\star, j^\star)$ which minimizes: $\left(\sum_{i \neq i^\star} \sigma_{i,j^\star} + \sum_{j \neq j^\star} \sigma_{i^\star,j}\right)$)
   (b) replace $(\sigma_{i,j})$ by the submatrix obtained by deleting row $i^\star$ and column $j^\star$
3. $\mu_{BM} = \sum_{k=1}^{\min(p,q)} \mu_k$ plays the same role as $|X \cap Y|$ in the classical Jaccard index, so we have: $J_{BM}(T,T') = \frac{\mu_{BM}}{p+q-\mu_{BM}}$

For example, we take $T = \{$`(1, 1, 2), (3, 5, 8), (13, 21, 34)`$\}$ and $T' = \{$`(1, 1, 2), (3, 6, 8), (12, 13, 14), (1, 1, 34)`$\}$. Since $T \cap T' = \{$`(1, 1, 2)`$\}$ only, we have $J(T,T') \approx 0.1666667$, while $J_{BM}(T,T') = 0.4$ because several items of T and T' are close.

A large number of similarity and distance measures can be used as well [15, 16]; actually, the method we propose is also suitable for frequently used measures, such as Ochiai [17] or Sørensen-Dice [18], which can be extended using the same best matching algorithm as we did for Jaccard. This point was checked experimentally through the same procedure as we present in section 5.

4.3 Transactions clustering

We apply the best-match Jaccard index for computing a distance matrix between all transactions in the database: $\Delta_{BM} = (\mathrm{d}_{i,j})$ with $\mathrm{d}_{i,j} = 1 - J_{BM}(T_i, T_j)$. The distance matrix can be used with a large number of clustering techniques; we chose a very classical hierarchical clustering algorithm, namely that implemented in the `flashClust` library in R [19], which easily provides dendrograms w.r.t. the similarity between transactions.

Then, the dendrogram can be cut into K classes (e.g. with the R function `cutree` [20]). The appropriate value of K is far from obvious, so we tried to evaluate empirically the appropriate height to cut the tree from randomly generated prototypes, where the value of K was known (see section 5). We found that a height $h \approx 0.6$ was quite convenient in all cases.

4.4 Prototype induction

Building a prototype for a class, means essentially finding a set of integer tuples (with zeros allowed) which has the highest matching value with the N transactions of the class w.r.t. the best-match Jaccard index. We start with a frequency analysis, i.e. for each item I that appears on some transactions of the given class, we compute $f(I)$ as the ratio between the number of transactions that include I and N.

Rare items, i.e. with $f(I) < \varepsilon$, may be considered casual purchase, or "noise", and simply discarded (typically this works fine with $\varepsilon \approx \frac{1}{N}$). Conversely, very frequent items, i.e. with $f(I) > \theta$, may be considered a must-have and kept unchanged in the prototype (typically $\theta \approx 0.95$).

Regarding intermediate items, we have to classify them again, in order to be able to detect that e.g. "SodaCola" products are always associated with "organic yoghurts" of several brands. Thus we compute a matrix distance between items: $(D_{i,j})$ with $D_{i,j} = 1 - \sigma(I_i, I_j)$ and use it for building a dendrogram of the items. There again, the number of classes K_I is not known a priori.

Therefore, we iterate the following process for several possible values of K_I:

1. for each cluster: build a prototype item by putting zeros where features differ ; for instance, if the items in the cluster are `(1, 5, 7)`, `(1, 6, 7)` and `(1, 12, 7)`, then the prototype item is `(1, 0, 7)`
2. collect all prototype items and join them with the very frequent items (see above) to build the candidate prototype P_{K_I}
3. compute the score of K_I as the average value of $J_{BM}(P_{K_I}, T)$ for all transactions T in the original class.

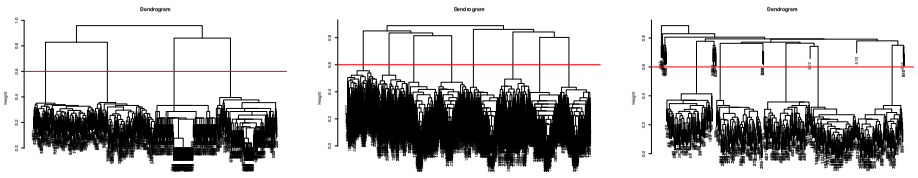
Finally, we keep the value $K_I^\star$ (and associated prototype $P_{K_I^\star}$) which maximizes this score.

5 Validation of the data mining process

As explained before, prototypes are used in a simulation process to produce artificial transactions. Since the agents perform autonomous behaviors, according to their shopping lists which contain prototype items (i.e. with wildcards), and in the situated context of a realistic store with possibly missing (or hard to find) items, the transactions that occur in the simulation are not expected to be exactly the same than the real ones.

Yet, we have to assess that the simulation is able to reproduce the same kind of customer behavior that is observed in reality. Therefore we can analyse the transactions produced by the simulation, build the corresponding prototypes through the same data mining process, and compare them to the prototypes that resulted from real data.

However, a first step towards analysing the outcome of the multiagent simulation is to prove the robustness of our prototype building method. Otherwise, possible differences between the prototypes that reflect the activity of the agents and the original ones, could not be explained by the behavior of the agents or the characteristics of the environment, but maybe just by a high sensitivity of the data mining process to perturbations. Thus we present here how the robustness of our analysis method has been tested.



(a) (b) (c)

Fig. 1. Dendrograms built according to the similarity of transactions in 3 experiments. Parameters: (a) $N_P = 4, N_I = \{5, 10, 20, 40\}, N_T = 200, N_A = 5\ \%, N_M = 5\ \%, N_O = 0$; (b) $N_P = 8, N_I = 10, N_T = 400, N_A = 5\ \%, N_M = 5\ \%, N_O = 0$; (c) $N_P = 5, N_I = 20, N_T = 100, N_A = 5\ \%, N_M = 5\ \%, N_O = 5\ \%$. The horizontal line represents the cut height (0.6).

5.1 Stochastic simulations of prototypes instantiation

In order to perform extensive tests, we generated several sets of prototypes, each one containing random prototype items. Then, in order to obtain a coarse-grained, but quick, simulation of the purchase induced by such prototypes, we instantiated them through a stochastic process, based on the following parameters:

- the number N_P of prototypes to test
- the number of prototype items $N_I(i)$ in each prototype i
- the number of transactions per prototype, $N_T(i)$, which determines how many transactions are instantiated for prototype i
- the number of additional random items $N_A(i)$ which indicates how many randomly chosen items are added to transactions of prototype i (we do so to represent e.g. casual or compulsory purchase which are not necessarily related to the prototypical purchase habits)
- the number of missing items $N_M(i)$ which indicates, in prototype i, how many prototype items will be kept uninstantiated (this offers the possibility that some items may not be found or not be needed)
- the number N_O of transactions that do not belong to any prototype (and generated completely randomly).

The "instantiation" of a prototype item consists in replacing each zero by a random, strictly positive, value. Thus this operation depends on the *domain* of the tuples that

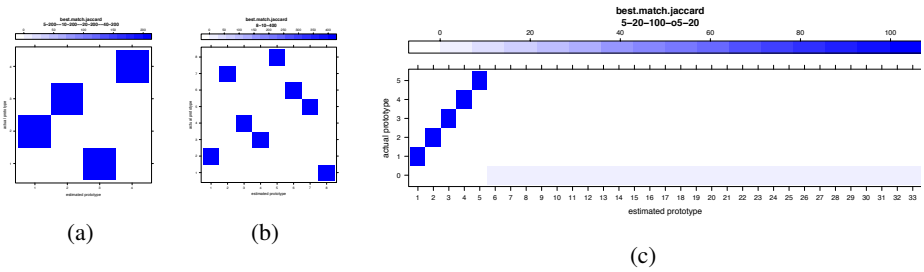
describe concrete items, i.e. the maximum values allowed for of each integer in the tuple. In our experiments we used 5-element tuples with maximum values `(20, 100, 10, 5, 2)` (on the basis of earlier work [11]).

We conducted automatic experiments and evaluations with a combination of all parameters within the following ranges: N_P: 4 – 10; N_I : 5, 10, 20, 40; N_T: 50, 100, 200, 400, 800; N_A: 0, 5, 10 % of items in the transactions; N_M 0, 5, 10 % of items in the transactions; N_O: 0, 5, 10 % of total transactions.

5.2 Results and discussion

As figure 1 shows on three experiments, transactions produced by the instantiation of random prototypes are well discriminated: cutting the trees at height ≈ 0.6 is sufficient to identify clusters that exactly reflect the original ones (cf. fig. 2a-b), even when transactions are built with random additional items or with random missing items.

When the database also contains random transactions, i.e. which do not come from any existing prototype, the clustering still identifies the original clusters, but also very small classes (see fig. 2c), which are most of the time singletons. When applying the induction process, those classes can be simply discarded because there is nothing to generalize in them. In all experiments (up to $N_O = 10$ % of the total number of transactions) the prototype building process was successful, which indicates enough robustness.



(a) (b) (c)

Fig. 2. Level plot comparison between estimated transactions clusters (abscissae) and original prototypes (ordinates) for experiments (a), (b) and (c). While (a) and (b) match perfectly, in (c) there are "noise" transactions (prototype "0") generated from pure random choice. It appears that they are *not* classified among the "true" clusters, but instead are put in small classes (here, 1 for each random transaction).

6 Experimental setup for multiagent simulations

At the present time, the multiagent simulator designed in [11] has been modified so as to represent shopping lists with prototypes and the items by integer tuples.

We have conducted preliminary experiments with the same randomly generated prototypes than in stochastic simulations. For now, the simulated transactions reproduce the same prototypes.

However, this result is dependent on the *time limit* which is given to the agents for their shopping trip. If too short, they exit the store without purchasing all items of their list. This does not affect the transaction clustering (because of its robustness towards missing items) but may alter the prototypes that are built from the simulated transactions. Indeed, the observation of the paths of the customers in the store points several "hot areas" where items are easily found: conversely, missing items are often the same (contrary to what happened in stochastic simulations), thus the corresponding prototype is a subset of the original one.

Far from being a limitation of the system, this property gives insights on how such kinds of simulation may help the placement of products, the spatial organization of the store, etc.: the limitation of time spent in the shopping trip is crucial, not only for the purpose of realism, but more significantly because it appears as the criterion that forces the store manager to optimize the positioning of products.

Ongoing work focuses now on the analysis and integration of large databases of real receipts. We are also modifying the environment in order to reproduce the store where the data were collected. We have for instance to integrate the actual positioning of items and information signs in the store. Indeed, the correctness of those informations may have a serious impact on the outcomes of the simulations, so we have to check them carefully and validate them with experts before we can start large-scale simulations.

7 Conclusion

The design of an integrated tool for decision support in the field of grocery retail and marketing is a long-term purpose indeed. However, in this paper we try to combine an incremental simulation approach (which is quite convenient to express complex hypotheses regarding individual behaviors, environmental configuration, etc.) with data mining algorithms (which usually indicates global, statistical features of a system). Our proposal is therefore able to endow agents populations with statistically realistic features, which in turn affect the behavior of the agents so as to produce statistically similar outputs. Far from being only qualitative, we show that this similarity can be measured. As shown above, our process is quite robust to noise in data. The results of the integration of real receipts in the multiagent simulation (still in progress) will be described in further publications.

Noteworthy, our method does not try to discover "true" classes of customers, such as a socio-economic, or demographic, or geographic segmentation would aim at. We only intend to capture similarities between traces left by individual actions, and use an abstract description of those traces as parameters of agents behaviors. Thus, taking into account geographic influences or seasonal variations merely relies upon an appropriate choice of the recording extent and duration for the real data.

Besides, we believe (though we cannot provide experimental evidences yet) that the transaction and prototype representation we propose can be applied to many other fields where the behavior of the entities can be characterized by such sets of features (e.g. molecular biology with phenotypical expressions of co-activated genes, ecology, or auction management), so as to participate in the bootstrap of multiagent simulations from empirical data.

References

1. Schwaiger, A., Stahmer, B.: Simmarket: Multiagent-based customer simulation and decision support for category management. In: Proceedings of MATES 2003: multiagent system technologies. Volume 2831 of LNAI., Springer (2003) 74–84
2. Siebers, P.O., Aickelin, U., Celia, H., Clegg, C.W.: Using intelligent agents to understand management practices and retail productivity. In: Proceedings of the Winter Simulation Conference (WSC'07), Washington, D.C. (December 2007) 2212–2220
3. Agrawal, R., Imielinski, T., Swami, A.: Mining association rules between sets of items in large databases. In: Proceedings of the 13th ACM SIGMOD Conference, Washington, D.C. (May 1993)
4. Mladenić, D., Eddy, W.F., Ziolko, S.: Exploratory analysis of retail sales of billions of items. In Goodman, A., Smyth, P., eds.: Frontiers in Data Mining and Bioinformatics. The 33rd Symposium on the Interface of Computing Science and Statistics. (2001)
5. Sheth-Voss, P., Carreras, I.E.: How informative is your segmentation? a simple new metric yields surprising results. Marketing Research (Winter 2010) 9–13
6. Zhang, T., Zhang, D.: Agent-based simulation of consumer purchase decision-making and the decoy effect. Journal of Business Research **60**(8) (August 2007) 912–922 Complexities in Markets Special Issue.
7. Yoann, K., Mathieu, P., Picault, S.: An interaction-oriented model of customer behavior for the simulation of supermarkets. In Huang, X.J., Ghorbani, A.A., Hacid, M.S., Yamaguchi, T., eds.: Proceedings of IEEE/WIC/ACM International Conference on Intelligent Agent Technology (IAT'10), IEEE Computer Society (September 2010) 407–410
8. Han, J., Kamber, M.: Data mining: concepts and techniques. Elsevier (2006)
9. Cavique, L.: A scalable algorithm for the market basket analysis. Journal of Retailing and Consumer Services **14**(6) (November 2007)
10. Cumby, C., Fano, A., Ghani, R., Krema, M.: Predicting customer shopping lists from point-of-sale purchase data. In: Proceedings of the 10th International Conference on Knowledge Discovery and Data mining (KDD'04), New York, NY, USA, ACM (2004) 402–409
11. Mathieu, P., Panzoli, D., Picault, S.: Virtual customers in an agent world. In Demazeau, Y., et al., eds.: Proceedings of the 10th International Conference on Practical Applications of Agents and Multi-Agent Systems (PAAMS'12). Advances in Soft Computing, Springer (2012)
12. Kubera, Y., Mathieu, P., Picault, S.: IODA: an interaction-oriented approach for multi-agent based simulations. Journal of Autonomous Agents and Multi-Agent Systems (2011) 1–41
13. Gibson, J.J.: The Ecological Approach to Visual Perception. Hillsdale (1979)
14. Jaccard, P.: Étude comparative de la distribution florale dans une portion des alpes et du jura. Bulletin de la Société Vaudoise des Sciences Naturelles **37** (1901) 547–579
15. Choi, S.S., Cha, S.H., Tappert, C.C.: A survey of binary similarity and distance measures. Journal of Systemics, Cybernetics and Informatics **8**(1) (2010) 43–48
16. Boriah, S., Chandola, V., Kumar, V.: Similarity measures for categorical data: A comparative evaluation. In: Proceedings of the SIAM International Conference on Data Mining (SDM 2008), SIAM (2008) 243–254
17. Ochiai, A.: Zoogeographic studies on the soleoid fishes found in japan and its neighbouring regions. Bulletin of the Japanese Society for Fish Science **22** (1957) 526–530
18. Dice, L.R.: Measures of the amount of ecologic association between species. Ecology **26**(3) (1945) 297–302
19. Langfelder, P., Horvath, S.: Fast R functions for robust correlations and hierarchical clustering. Journal of Statistical Software **46**(11) (2012) 1–17
20. Becker, R.A., Chambers, J.M., Wilks, A.R.: The New S Language. Wadsworth & Brooks/Cole (1988)