Blundo *et al. Hum. Cent. Comput. Inf. Sci.* (2017) 7:26
DOI 10.1186/s13673-017-0107-4

Human-centric Computing and Information Sciences

RESEARCH Open Access

An AmI-based and privacy-preserving shopping mall model

Carlo Blundo[†], Francesco Orciuoli[*†] and Mimmo Parente[†]

*Correspondence:
forciuoli@unisa.it
[†]Carlo Blundo, Francesco Orciuoli and Domenico Parente contributed equally to this work
Dip.to di Scienze Aziendali-Management & Innovation Systems, Università degli Studi di Salerno, Via Giovanni Paolo II, 132, 84084 Fisciano, SA, Italy

Abstract

Nowadays, large shopping malls provide tools to help and boost customers to buy products. Some of these tools melt down digital operations with physical ones executed by customers into blended commerce experiences. On the other hand, ambient intelligence (AmI) represents a paradigm focused on equipping physical environments to define ergonomic spaces for people interacting with computer-based localized services which are ubiquitously accessible. In this context, we propose a framework based on cellular automata (CA), a very well known formal computational model, suitable to abstract services deployed into an AmI-based environment preserving certain privacy levels of shoppers' information. CA-based algorithms are advantageous because they are distributed, scalable, on-line and require low costs to be deployed. This work proposes a recent application of CA, namely Cellular ANTomata, to implement a service by which shoppers are guided to find the suitable offerings for items in their shopping lists. A further result provided by this paper is the instantiation of a protocol for privacy-preserving shopping experience in the shopping mall.

Keywords: Ambient intelligence, Cellular automata, Privacy, Blended shopping

Introduction

Blended Shopping is defined by the authors of [12] as the "execution of the transaction phases (information, mediation, negotiation, contracting, fulfillment, and after-sales) involving both, real sales and presentation mechanisms as well as network based sales functionality". The idea of blended shopping is borrowed from the concept of multi-channel behaviour of customers, who use different distribution channels for the same purchase. It is a common behaviour to taste a product in a physical store and buy such product through an e-commerce site in order to obtain a lower cost. Blended shopping enables merchants to offer services and information best fitting to the consumer's needs, resources, and situation and this leads to higher customer satisfaction. Moreover, the merchant who offers "information and advice aims to clinch the deal instead of loosing it to e-Commerce competition" [12].

Of course, the precondition for creating blended shopping scenarios is that merchants should be able to run both interlinked channels: physical and digital. Hence, we believe that shopping malls are right environments to implement and execute blended shopping scenarios. Shopping malls can be technologically enriched and transformed into intelligent ambients by employing the ambient intelligence (AmI) paradigm where customer

experience is significantly improved in terms of interactions with the services (payment, product search and browsing, couponing, etc.) and cost saving and, on the other hand, merchants can effectively and efficiently deliver such services and be competitive. Furthermore, it seems that the technologies (e.g., mobile computing, cloud computing, sensor networks) for ubiquitous computing (with specific reference to u-commerce) [35] can be effectively employed for blended shopping.

AmI can be defined as a digital environment that supports people in their daily lives by assisting them in an intelligent way [34]. AmI systems have concrete environments and real occupants who interact with them. Moreover, an AmI system must be "intelligent", i.e., it has to intervenes only when needed, and has to adapt its behaviour to current overall situations, users' preferences and needs, and so on. In this context, the authors of [13] claim that AmI is the right opportunity to construct blended shopping ecosystems.

Furthermore, the authors of [11] affirm that success of AmI will depend on how privacy and other rights of individuals can be protected and how individuals can come to trust the intelligent ambient that surrounds them and through which they move, see [14]. In the same work, the AmI-based shopping application domain is presented as a context in which privacy issues are crucial.

In light of the above considerations, this work proposes the definition of a (location-based), see [20], context-aware recommender system, see [19], providing the main functionalities of an indoor navigation system (INS), see [8], deployed in a shopping mall. The goal of such system is to give, step-by-step, the right indications to a shopper (within a shopping mall) to allow her arriving to the shop that proposes the most convenient offering/product (e.g., lower cost in the mall) for items in her wishlist. The system implements privacy mechanisms for both shoppers and shops.

Scenario

Nowadays, it is usual to see shopping malls or big stores proposing weekly or daily offerings (for limited stocks of specific products). By using mechanisms like, for instance, magazines distributed at the entrance of the shop or Web sites. There are two main drawbacks with these traditional tools. In fact, the information shared they provide is not contextualized and cannot be regulated by some adaptation processes. For example, if the stock for a specific product, that is associated to an offering, is finished then it is not possible to suddenly and agilely replace this offering with another one. Unfortunately, the above described capability is desirable because it can enable more flexible marketing strategies that can be adjusted on-the-fly, also by considering the current request (how many shoppers are interested in some kind of product?). In order to provide this capability it can be possible to reason on two facts. The first one is that smartphones (and other smart devices) are widespread, so it is reasonable to think that shoppers can receive information about the current offerings while they are in the mall. The second one is that it is possible to empower closed physical environments by using low-cost sensors/actuators exploited (in the context of wireless sensor networks) to sense *presence*, detect *location* and know *desires* (in the form of wishlists) of the shoppers and *suggest* them how to move (*recommend* the next move) in the shopping mall to reach and gather offerings to be exploited during the purchase. Both facts contribute to model an *intelligent ambient* whose objective is to guide shoppers toward the current best offerings

related to the products they really need (indicated on their *wishlist*). We assume that offerings as well as products are localized in the shops which provide them. So, shoppers have to reach these shops to gather offerings and perform the product purchase. The physical environments we are considering are those buildings in which there are numerous shops owned and managed by different merchants. This scenario provides some clear benefits for shoppers. In fact, they are guided to reach offerings and products they really need and can obtain lower costs for the items they would buy. On the other side, merchants can have the tool to achieve single shoppers and propose only the offerings they are interested in. In case that more merchants provide different offerings for similar products, highest priority is given to the lowest-cost offering. In this way the merchants proposing best offerings are rewarded because they are visible to the shoppers earlier than other merchants.

As anticipated by the previous section, a real-world AmI application, especially in the shopping domain, should consider privacy issues. In the proposed scenario, privacy is considered from several viewpoints: empowerment and regulating agent, see [11]. The first one implies that people should have the power to control the publication of personal information like, for instance, the details on his/her wishlist items. The second one should ensure balance among players within a competitive environment. This is exactly the case of merchants that knowing lowest-cost products of their competitors could maliciously act in order to win the competition.

Related works

Similar works are recognizable in literature. The authors of [22] propose an indoor location-based recommender system aiming to recommend the shops in the mall that will interest the customers according to their track history. Over 10,000 customers are tracked for a week in the Bow Valley Square shopping mall. This data is used to develop a recommender system to predict the preferences a user will give to all the shops based on the data set and recommend the top shops to the user. A virtual assistant for shopping mall is defined in [36]. Such system is mostly focused on human-computer interaction issues and provide a user interface based on projector phones applying concepts of augmented reality. Projected interfaces offer additional distinct advantages over static guides and even traditional or augmented reality mobile applications. The authors describe five concepts for a shopping mall indoor assistance system based on projector phones, comprising support for shop selection, precise way finding, "virtual fitting" of clothes, and context-aware and ambient advertisements. Moreover, in [37] it is proposed a location-aware recommender system that accommodates a customer's shopping needs with location-dependent vendor offers and promotions. The authors of [27], present the system SugarTrail for indoor navigation assistance in retail environments that minimizes the need for active tagging and does not require existing maps. By leveraging the structured movement patterns of shoppers in retail store environments, the system provides higher accuracy than existing radio finger-printing approaches. In [25] the authors propose a work to demonstrate how image content can be used to realize a location-based shopping recommender system for intuitively supporting mobile users in decision making. Generic Fourier Descriptors (GFD) image content of an item were

extracted to exploit knowledge contained in item and user profile databases for learning to rank recommendations.

With respect to the aforementioned related works, our system offers several advantages. First of all, [22] and [37] do not provide INS features. Conversely, [27] provides only navigation features. Lastly, [36] and [25] mostly focus on augmented reality and image analysis respectively. All these works center their attention on some specific aspects. They do not try to provide a solid framework on which building application scenarios. This is exactly what we have done by adopting cellular automata that enable a decentralized management of information, providing adaptation capabilities to the whole system. Lastly, privacy mechanisms provide added value to our proposal with respect to the works recognizable in the existing literature.

The intelligent shopping mall

The intelligent ambient we consider is a shopping mall equipped with specific hardware for supporting the functionalities of the proposed context-aware recommender systems. W.l.o.g., we consider only one floor shopping mall, the case with more floors can be dealt with analogously. The shopping mall floor can be seen as divided into zones and each zone can be a shop, a rest area, a corridor, etc.

Figure 1 shows a view of the ambient. According to the principles of ambient intelligence (AmI), environments must be unobtrusive, interconnected, adaptable, dynamic, and intelligent [31]. In such environments, traditional computers, input and output devices disappear and are replaced by processors and sensors, integrated in everyday objects (e.g. clothes, household devices, furniture). The envisioned AmI environment is responsive to the needs of its inhabitants and it is aware of their personal requirements

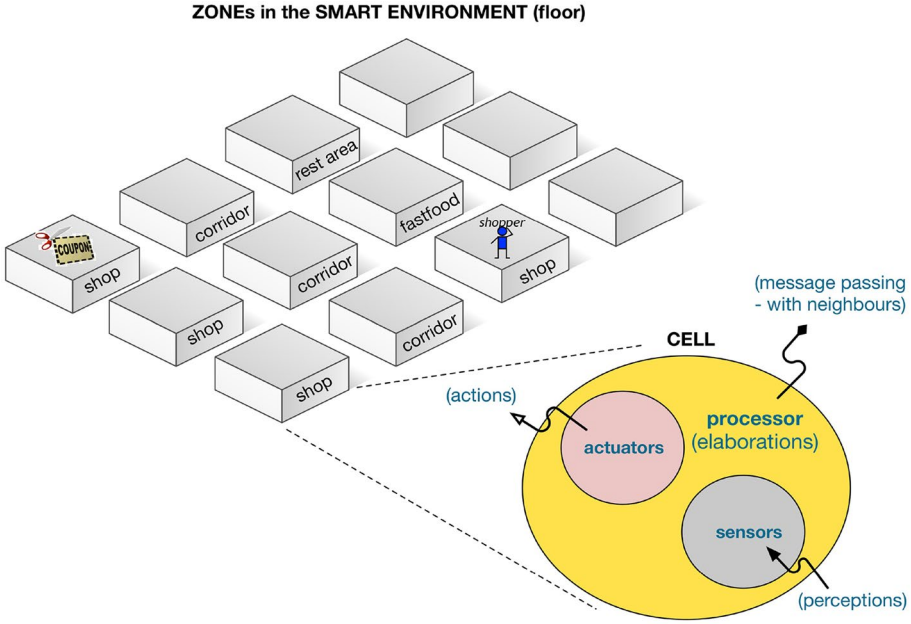

Fig. 1 Conceptual view of the intelligent shopping mall

and preferences and interacts with people in a user-friendly way [31]. We map (model) each zone with a *cell*. Each cell is equipped with the following elements:

- *Sensors*, employed to "observe" the environment and its inhabitants (*shoppers*). In particular, sensors are used to perceive both the presence of shoppers in that zone and/or active available offerings (if the zone is a shop).
- *Actuators*, employed to communicate with the shoppers of the environment. Actuators are output devices useful to send recommendations to the shoppers in that zone.
- *Processors* elaborate input (from sensors) and generate output (communications for shoppers and for neighbor cells).

The AmI model we propose in this work is borrowed from [30] that provides a five-layers model: (1) sensors and actuators, (2) AmI networks and middleware, (3) devices, (4) service, and (5) applications. The second layer is the most important and provides the ambient with intelligent capabilities. In the present work, the intelligence of the ambient is completely distributed according to the cellular automata (CA) model that provides simplicity (the basic processing element, the cell, is simple), locality (all interactions take place on a purely local basis, a cell can communicate with a few other cells), scalability (it is easy to upgrade a CA by adding other cells or changing its topology) and robustness (CA continue to perform even when a cell is faulty because the local connectivity property helps to contain the error). In particular we will use the Cellular ANTomata, an extension of cellular automata, introduced in [28, 29] as a model for realizing ant-inspired algorithms that coordinate robots within a fixed, geographically constrained environment.

Cellular automata and Cellular ANTomata

Cellular automaton (CA) consists of a regular *network* of *extremely simple computers*, (called *cells*) which are essentially finite state machines (FSMs). They have been studied since early ′60s and are still investigated mainly because they combine a mathematical simplicity and elegance with an high level of computational efficiency and efficacy that makes them very suitable to implement real case scenarios [16–18, 23, 33]. The cells operate synchronously, at discrete time unit. At each time t, a *configuration* specifies the state of each cell. Time is discrete, and at each time step each cell is in one of a finite number of *states*. A *neighborhood* relation is defined, indicating the *neighbors* of each cell. All the cells have the same number N of neighbors, except a fixed number of *boundary cells* which have less neighbors (throughout our paper $N = 8$). A cell is intended to be linked to each of its neighbors through *communication channels* and can send and receive, at each time step, *messages*), which are binary sequences whose length is bounded by the constant capacity of the channels. At each time step, every cell updates its state in accordance to a *state-transition function* δ that takes as input the state of the cell itself along with the messages received from the cells in its neighborhood. A *computation step* modifies the configuration, in accordance to the transition function and depending on both the current configuration and the sequences sent by the cells. An *initial configuration* is a configuration at time 1. Observe that in the classical definition of CA, the transition function takes as input the state of the cell itself and those of its

neighbors at the previous step. This classical definition is captured here when the capacity of the channels is $log|Q|$, where Q is the set of states of the CA, thus each cell can send its whole state in a single step.

In [28, 29] the author proposes the definition of a set of algorithms running over the Cellular ANTomata and, among them, the Food Finding algorithm was described that, due to its peculiar characteristics, allows us to provide an efficient and effective solution for our Intelligent Shopping Mall scenario. The ANTomata are classical cellular automata with the feature that each cell is equipped "with sensor for ants, obstacles and goals and with a unidirectional communication channel that a resident ant will respond to". In this way, the messages flow through the network "below the surface that objects (ants, obstacles and food) reside on". In our scenario the shopper plays the role of the ant whose goal is to reach the products listed in her wishlist.

Sketch of the Cellular ANTomata algorithm

Recall that we have established a map among physical zones (in the shopping mall) and cells (in the CA). If a shopper is in a specific zone, the corresponding cell in the model is aware of this presence by means of the *sensors* occurring in the zone, the cell can elaborate this information and produces results by means of its *processor* and, lastly, it is able to interact with such shopper by using its *actuators*. Thus, a shopper is localized at a specific zone in the mall and communicates her wishlist to the corresponding cell. The cell has the task to recommend the next move to the shopper to let her reach the zones selling products that match one or more items in the shopper's wishlist. Once the shopper received the recommendations, she can follow one of them or she is free to move autonomously in the mall. Recommendations are represented by single steps toward the next zone to reach. These can be graphically presented to the shopper as proposed in Fig. 2 where the shopper receives the first recommendation that invites her to go ahead in the corridor and reaching zone c_x. When the shopper has moved to c_x, she receives a

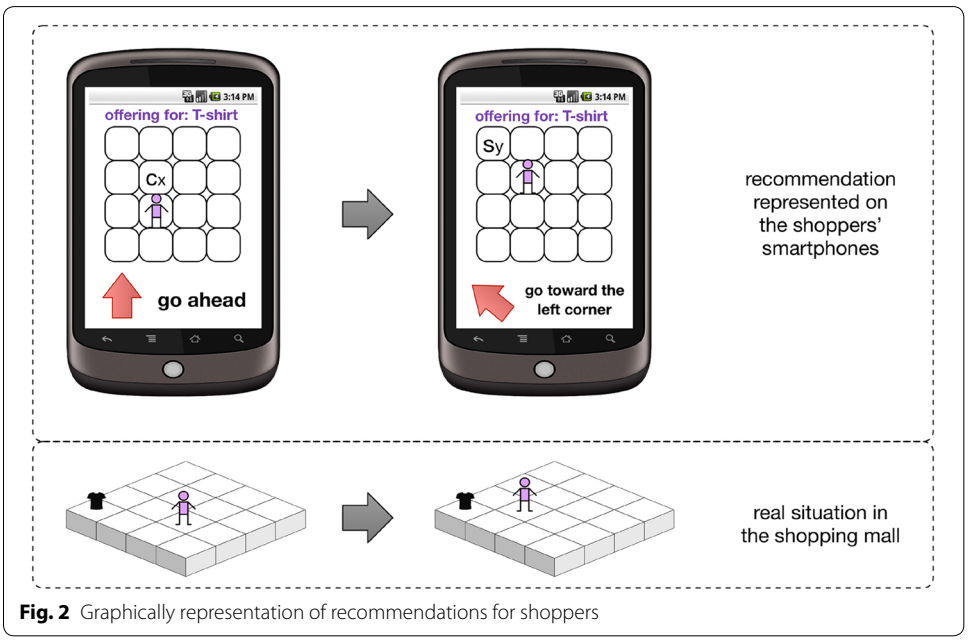

Fig. 2 Graphically representation of recommendations for shoppers

new recommendation inviting her to go toward the left corner to reach shop s_y. Labels c_x and s_y are zone identifiers that can be also replaced by intelligible names as shop names that can be simply recognized by the shoppers.

Thus, shoppers need the recommendation and some markers (e.g., zone names) in the environments in order to move according to the Intelligent Shopping Mall. A special kind of recommendation invites the shopper to buy a product (matching one in the wishlist) in the zone where she currently is. Of course, this zone is a shop.

Recommendations are generated by means of a distributed algorithm running over the CA. Such an algorithm, resembles the Food-Finding-Algorithm of [29]. The shoppers in the mall are the ants within the CA. Shoppers look for products (with suitable costs) in the mall as ants look for food. In particular, in order to recommend a direction to a shopper for products in her wishlist, each cell (a zone in the mall) must match the items in the shopper's wishlist with the list of products which are in the shop or in case there are none (or the cell is associated to a zone that is not a shop), what is the next move to reach the products in the mall. The former point is satisfied by means of the sensors gathering information from the shoppers smartphones. The latter point is accomplished by exchanging a particular directive (sometimes we call it also *piece of information*) called I-HAVE-PRODUCT, over the CA. The message exchange algorithm is described in detail in "Implementation details of the Cellular ANTomata algorithm". Informally the algorithm works as follows: every cell in the CA selling a product sends a message to all their neighbours at each time tick (step), repeatedly. The message provides information about both the product and the direction to follow in order to reach it, this information shall be available at the next time step. Figure 3 shows two of the eight messages informing the neighbours that node x has a product P and how to reach it. Each one of the eight messages is contextualized by considering its destination. In fact, the message for y invites to follow the direction *SW* (`south-west`).

Until the shop x sells its product P to some shopper, the I-HAVE-PRODUCT directive is broadcasted by the shop at each subsequent time step. In order to reach also cells/zone that are not neighbors of a broadcaster, a propagation mechanism is needed. The idea is that once a cell receives a message it must *relay* this directive to a subset of its neighbors.

Figure 4 shows the RELAY operation. In particular, the original directive started from x at time 1, will arrive at y at time 2 and from here it is relayed. Thus, at time 3 the relayed directives arrive to a, b, and c (actually, as we will see when the algorithm is detailed, these two directives will be merged in a unique message).

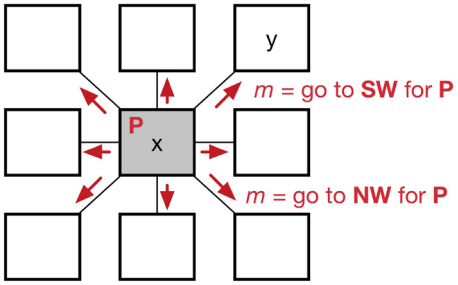

Fig. 3 Examples of I-HAVE-PRODUCT message exchange (broadcast)

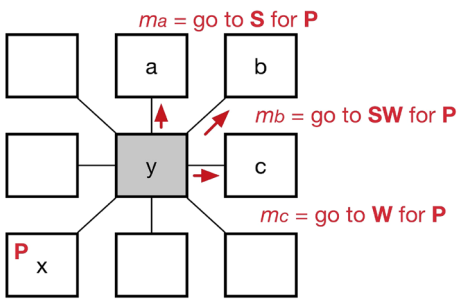

Fig. 4 Examples of RELAY scheme

By using this approach, after a number of iterations (steps), each cell is aware of the directions to recommend to possible shoppers for all the products managed by the intelligent system.

Technological and architectural issues

In order to implement the Intelligent Shopping Mall based on the Cellular Automata model we need to concretize the concept of cell. A cell can be implemented by using a low-cost micro-computer (e.g., Raspberry Pi2 single-board computer[1]) that can be equipped with a sensor to detect the presence of shoppers and receive their wishlists, an actuator to send recommendations to the shoppers, and a wireless mechanism to communicate with its neighbors. Such a micro-computer is installed at each zone of the mall. The sensor device can be realized by using a Bluetooth or other alternatives (low-energy Bluetooth, ZigBee [32]). The idea is that the sensor over the micro-computer detects the smartphone of the shopper that is in the zone and, thanks to a specific, but simple App running on the phone, it is able to ense the presence of the shopper, identify her and read her wishlist. Sensing the presence and identifying a shopper in a zone are task known as localization. Localization methods in Indoor Navigation Systems can be grouped into four different techniques [8]: (1) dead-reckoning, (2) direct-sensing, (3) triangulation, and (4) pattern-recognition. The proposed approach is based on a direct-sensing localization task. The Bluetooth is used also as an actuator when the cell (micro-computer) has to send recommendation to the shopper in the zone. In the proposed Cellular Automata model, each cell has eight neighbors, thus each micro-computer has to be connected to eight micro-computers. Hardware like Raspberry Pi2 can be equipped with WiFi dongles enabling networking among them. The cells can be easily programmed by using Java, Python or any other programming language compatible with the Linux-based Operating System installed on the Raspberry Pi2.

Implementation details of the Cellular ANTomata algorithm

The distributed algorithm is executed by a set of interconnected simple Finite Automata (FA) deployed into a mesh Γ of cells, modeling the smart shopping mall. Let $\mathcal{S}$ be the set of cells that are shops in Γ and let $\mathtt{P}$ be all products sold in the shopping mall.

[1] https://www.raspberrypi.org/products/raspberry-pi-2-model-b/.

Each cell c has eight neighbors $N(c) = \{c_N, c_{NE}, c_E, c_{SE}, c_S, c_{SW}, c_W, c_{NW}\}$ and, at each time transition, it can communicate with them by exchanging messages. In a pre-processing phase of the algorithm, the central authority establishes the set of products $\mathtt{AP} = \{p_1, \ldots, p_n\} \subseteq \mathtt{P}$ that are recommended (available) to the shoppers (anyway during the execution of the algorithm this set may vary, because some products can be bought). Let n denote the number of available products and $\mathtt{AP}_c \subseteq \mathtt{AP}$ be the set of products available in the shop $c \in \mathcal{S}$.

Computation goes on through fixed time steps. At each time step, each cell c receives (reads) a message from each of its neighbors in $N(c)$ indicating the presence of products at each cardinal direction. The cell c knows the products it owns (i.e., $\mathtt{AP}_c$) and reads the wishlists of the shoppers detected in the zone associated to c. In particular the cell c produces:

- Messages for all its neighbors informing them if there are products available in it or reachable through it, and
- Recommendations for each shopper in the zone associate to c for the products in her wishlists available in that zone or reachable by moving in one of the eight cardinal directions.

A shopper, localized at the zone associated to c, communicates her wishlist to c, receives recommendations from c and can purchase products owned by c. At each time step, all the cells change state according to their current state and the messages received by the neighbours. At the next time step this computation at each cell is repeated. The whole algorithm ends when the set of all available products $\mathtt{AP}$ is empty or there are no shoppers in the mall. Later on, we will detail how this is accomplished.

Wishlist and state representation

Let n be the number of products available in the mall (i.e., $n = |\mathtt{AP}|$). Any shopper entering the mall is interested in buying the products in her wishlist $\mathtt{WL}$ that we represent by a characteristic vector $w = b_1 b_2 \ldots b_n$ of n bits denoting the occurrence of the products in $\mathtt{WL}$, that is

$$b_i = \begin{cases} 1 & \text{if } p_i \in \mathtt{WL} \\ 0 & \text{if } p_i \notin \mathtt{WL}. \end{cases}$$

Recall that each cell c is essentially a finite automata and a generic state q of c at a given time t depends on the products available in the cell c and the messages received from his neighbors $N(c)$. Such a state is coded as

$$q = [q_C, q_M]. \tag{1}$$

Notice that the state q should depend on the cell c and the time t, as well. Since in the following no ambiguity arises, to avoid overburdening the notation, we omit both c and t. The first component q_C of a generic state q is a characteristic vector of n bits representing the occurrence of the products in $\mathtt{AP}_c$, that is

$$q_C = b_1 b_2 \ldots b_n \tag{2}$$

where

$$b_i = \begin{cases} 1 & \text{if } p_i \in \texttt{AP}_c \\ 0 & \text{if } p_i \notin \texttt{AP}_c \end{cases} \tag{3}$$

The second component q_M of a generic state q is an 8-dimensional vector representing presence/absence of a given product towards c's eight neighbors $N(c) = \{c_N, c_{NE}, c_E, c_{SE}, c_S, c_{SW}, c_W, c_{NW}\}$. More in details,

$$q_M = \begin{bmatrix} q_M^N \\ q_M^{NE} \\ q_M^E \\ q_M^{SE} \\ q_M^S \\ q_M^{SW} \\ q_M^W \\ q_M^{NW} \end{bmatrix} \tag{4}$$

where each q_M's component is an n-bit characteristic vector. For example, in

$$q_M^N = m_1 m_2 \ldots m_n \tag{5}$$

$m_i = 1$ (resp., $m_i = 0$) indicates that product p_i occurs (resp., does not occur) towards `north` of c.

Initially, a synchronization phase is needed to inform all the cells of the mall that the algorithm starts, see [16, 18]. The synchronization phase makes the automaton of each cell transiting from the inactive state to the initial state. Initially, $\texttt{AP}_c = \emptyset$ for all $c \in S$ and the inactive state is coded, for all cells, with an all-zero bit-string. After the synchronization phase, all the $\texttt{AP}_c$ are initialized. Therefore, the initial state for a cell $c \in S$ depends on $\texttt{AP}_c$. For instance, let $\texttt{AP} = \{p_1, p_2, p_3, p_4\}$ and suppose that $\texttt{AP}_c = \{p_1, p_3\}$. Then, the initial state for c is coded as follows:

$$q = \begin{bmatrix} 1010, & \begin{bmatrix} 0000 \\ 0000 \\ 0000 \\ 0000 \\ 0000 \\ 0000 \\ 0000 \\ 0000 \end{bmatrix} \end{bmatrix} \tag{6}$$

The initial state q indicates that cell c intends to broadcast that it owns products p_1 and p_3. Suppose now that c has received a message from `northwest` indicating the presence of product p_2 (in the example, $q_M^{NW} = 0100$). Then, according to the algorithm sketched in the previous section, it has to relay this information to `south`, `southeast`, and `east`. Moreover, suppose there is one shopper in the cell c that needs the products p_1 and p_2. Her wishlist is coded as a 4-bits characteristic vector, in this example

$w = 1100$. The first product is present in c and the second one is reachable by moving to `northwest`. The recommendation in c for the shopper is composed by two parts by using w and the state q as follows: first a bitwise AND operation is performed between the wishlist w and the first component q_C of q, to let the shopper know about the availability of products in c (in the example, p_1); then, another bitwise AND operation between w and each dimension in the second component q_M of q is computed to indicate the direction to go for the other product (in the example, the AND between w and q_M^{NW} specifies that the product p_2 can be found by moving to `northwest`).

Transition function

In this subsection we explain the behavior of the transition function δ at cell c, by which each cell at each time step changes its current state in a new state.

First, the set $\texttt{AP}_c$ is updated in case some shoppers bought products from c by setting to zero the bits of the corresponding items of the first component of the state (see next subsection of the shopper representation). Assume, for the sake of the simplicity, that if a shop c has any product sought by a shopper, then such a shopper buys the product from c. Then, the second component of the new state is composed by merging the information of the I-HAVE-PRODUCT and of the RELAY scheme as follows. The I-HAVE-PRODUCT scheme is very simple and was explained by an example in Fig. 3. All cells associated to shops owning a recommended (available) product broadcast the information I-HAVE-PRODUCT to all neighbors. The RELAY scheme, presented by an example in Fig. 4, is fully detailed in Fig. 5. Additional issues on the RELAY scheme will be presented in "Relay scheme issues". Since the information of both schemes are n-tuples of bits, then the messages to send are effectively composed by computing a bitwise OR between the strings representing the messages of the schemes. The messages arriving from adjacent cells are n-tuples indicated as $\mathbf{m}sg_N, \mathbf{m}sg_{NE}, \ldots, \mathbf{m}sg_{NW}$, where $\mathbf{m}sg_Y$ is the message received at the previous step from direction Y.

Formally, the transition function δ has as input the current state $q = [q_C, q_M]$ of the cell c and an 8-tuple constituted by the messages *received* by c's neighbours. Hence,

$$\delta(q, (\mathbf{m}sg_N, \mathbf{m}sg_{NE}, \ldots, \mathbf{m}sg_{NW})) = p. \tag{7}$$

The new arriving state p is of course of the form $p = [p_C, p_M]$: where p_C is computed by updating $\texttt{AP}_c$ as illustrated before (i.e., $p_C = q_C \wedge \overline{w}$, where $\overline{w}$ denotes the items of `AP` not occurring in w);[2] while the second component p_M of the state p, is computed as follows:

$$p_M = \begin{bmatrix} p^N = \mathbf{m}sg_N \\ p^{NE} = \mathbf{m}sg_{NE} \\ p^E = \mathbf{m}sg_E \\ p^{SE} = \mathbf{m}sg_{SE} \\ p^S = \mathbf{m}sg_S \\ p^{SW} = \mathbf{m}sg_{SW} \\ p^W = \mathbf{m}sg_W \\ p^{NW} = \mathbf{m}sg_{NW} \end{bmatrix} \tag{8}$$

[2] We are assuming that there is only one unit of any given product. Obvioulsy, this is too restrictive, but it will simplify the description. If a shop has more than one unit of a product, it just will decrement a counter when selling it. The first component p_C of the state p will be updated when the product-counter reaches zero.

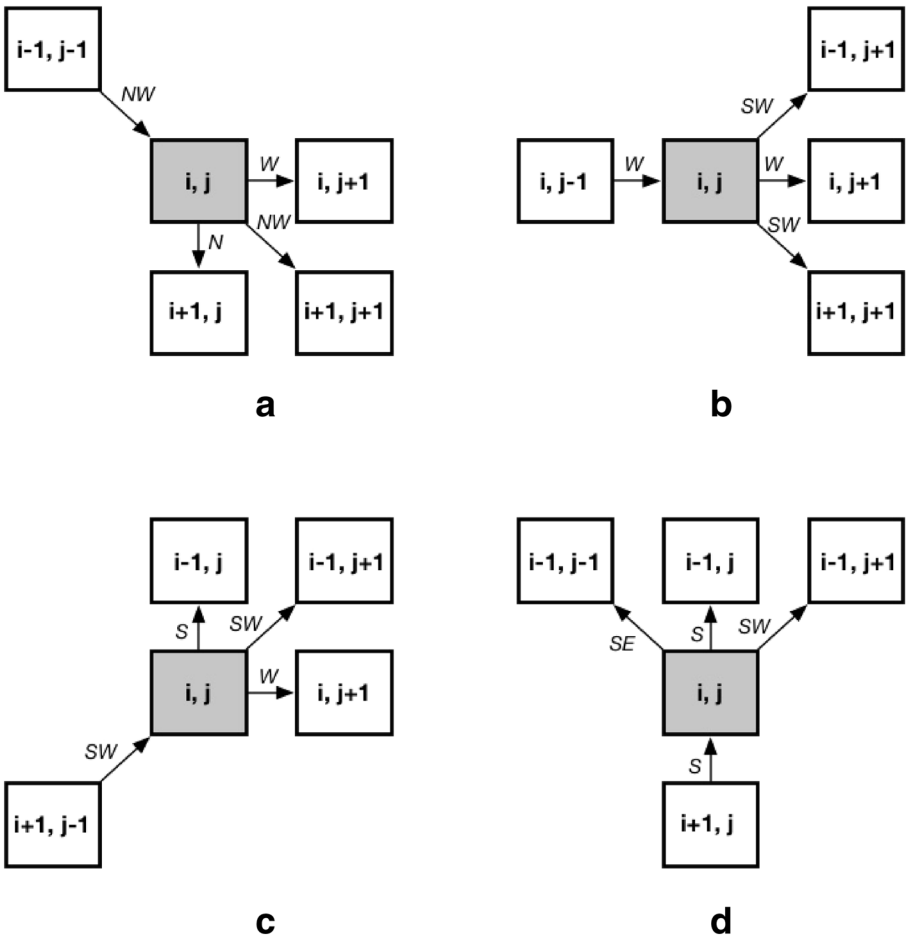

Fig. 5 RELAY scheme at node (*i*, *j*)

where, for example, denoting by $r = [r_C, r_M]$ the state of the cell at time t at the south of c,

$$\mathbf{m}sg_S = r_C \vee r_M^{SW} \vee r_M^S \vee r_M^{SE}. \tag{9}$$

Other messages are computed accordingly.

ANT/Shopper representation

In the proposed model, the shoppers in the mall represent the ants of the Cellular ANTomata, looking for foods. Shoppers receive recommendations from the cell associated to a zone in which they are sensed. Such recommendations are generated (by cells) by using both the messages they receive by neighbours and the set of products $\mathtt{AP}_c$ they directly sell (assuming the computation at the cell c).

Thus, let us define $\mathbf{R}_w$ as the recommendation for a generic shopper with respect to the products in her wish list w. In particular:

$$\mathbf{R}_w = w_R \mathbf{R}^N \mathbf{R}^{NE} \ldots \mathbf{R}^{NW} \tag{10}$$

where $w_R = b_1 b_2 \dots b_n$ is an n-bit vector representing the products in $\texttt{AP}_c$ that also appear in the shopper wishlist w computed as the bitwise AND between q_C and the vector w. Hence,

$$b_i = \begin{cases} 1 & \text{if } p_i \in \texttt{AP}_c \text{ and } p_i \in \texttt{WL} \\ 0 & \text{if } p_i \notin \texttt{AP}_c \text{ or } p_i \notin \texttt{WL}. \end{cases}$$

The other components of $\mathbf{R}_w$ represent the recommendation to the shopper for the products not in $\texttt{AP}_c$. Without loss of generality, we will show the format for the products reachable moving one step to `north` (i.e., $\mathbf{R}^N$) the other ones being similar.

$$\mathbf{R}^N = r_1^N r_2^N \dots r_n^N \tag{11}$$

where

$$r_i^N = \begin{cases} 1 & \text{if } c \xrightarrow{N} p_i \\ 0 & \text{if } c \overset{N}{\nrightarrow} p_i \end{cases}$$

In this case, $c \xrightarrow{N} p_i$ ($c \overset{N}{\nrightarrow} p_i$, resp.) means that p_i can (cannot, resp.) be reached by moving to the `north` starting from c. $\mathbf{R}^N$ can be computed by applying the bitwise AND operator between q_M^N and w.

Correctness

The correctness of the algorithm derives from the proof for the *Food Finding* algorithm provided by Rosenberg in [28, 29]. Without loss of generality and in order to provide a clear description we assume that: (1) we have r shoppers interested in the same product type for which we have s available offerings, (2) each of these shoppers is interested in only one item related to the above mentioned product type, and (3) the shoppers have only one product in their shopping lists. If $r \geq s$ then every shopper will eventually follow a message that ends in the cell containing the offering. Shoppers are considered inactive (with respect to the algorithm) if they physically leave the mall or if their shopping lists are empty or do not match with at least one active offering. Otherwise, they are considered active shoppers. If $s < r$ then some shoppers will eventually reach every offerings. The assumptions introduced before does not impact on the generality of the problem. In fact, if a shopper is interested in, say, 3 products of the same type it is possible to consider her as three different shoppers. Moreover, this proof can be applied u times for u product types in order to consider all the types of products in the mall. Termination is guaranteed by cell (1, 1) that continues to poll to determine if the stop criterion (no active shoppers or no active offerings in the mall) is reached. Lastly, as well as the *Food Finding* algorithm, the Offering Finding process terminates $O(r \cdot \max(k, m))$ steps after either there are not active shoppers in the mall or there are not active offerings, where the size of the mall is $k \times m$.

Virtual pheromones

Consider a shopper having in her list products of some type t. When she is present in a cell, it may happen that she receives two `I-HAVE-PRODUCT` directives for type t products coming from different directions, then our algorithm "recommends" in a

clockwise way starting from the `north`, the direction to the shopper. Anyway it is possible to employ a nature-inspired mechanism, namely the *pheromone* that is a volatile organic compound produced by ants. *Virtual pheromones* are introduced in [28, 29]. Virtual pheromones are useful to enable cells suggest shoppers the shortest path when more than one direction can be followed at a given iteration. In particular, the cells in the mesh maintain a finite number of counters, say c, whose ranges are $[0, l_1], \ldots, [0, l_c]$. Each counter is associated to a product type for which an active offering is present. At the beginning of the algorithm, the cells having offerings to broadcast, set their counters to the maximum intensity level of pheromones. In the next iteration the broadcasted messages are relayed by the cells which have received the original messages. Such cells decrease of one unit the intensity level of pheromone. If more than one message of the same type is received with different levels of pheromones, the intensity is set to the *maximum* between the two and then decreased by one.

Relay scheme issues

By fully adhering to the original algorithm of [29], the scheme illustrated in Fig. 5 does not guarantee the correctness of the algorithm. In fact, once an offering message has been generated, it will continue looping indefinitely also when the corresponding offering has been picked up by a shopper. As shown in Fig. 6, once a message is generated (gray squares), after eight relay iterations it performs a first loop and continue to circulate indefinitely causing an inconsistent behaviour of the algorithm.

A simple approach to avoid the aforementioned unwished behaviour consists in preventing that the same message moves for two consecutive times in a clockwise direction or two consecutive times in a counter-clockwise direction.

Figure 7 shows an example of the adjusted *relay* scheme avoiding the unwished behaviour. In particular, assume that cell (4, 2) contains an offering and, thus, broadcasts the

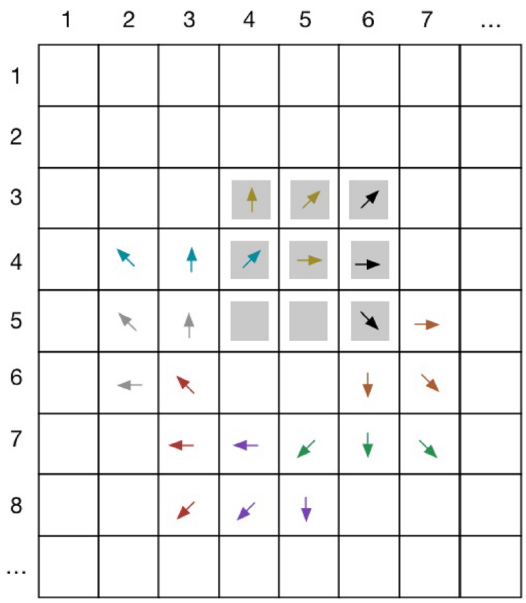

Fig. 6 Infinite loop generated by the *relay* scheme

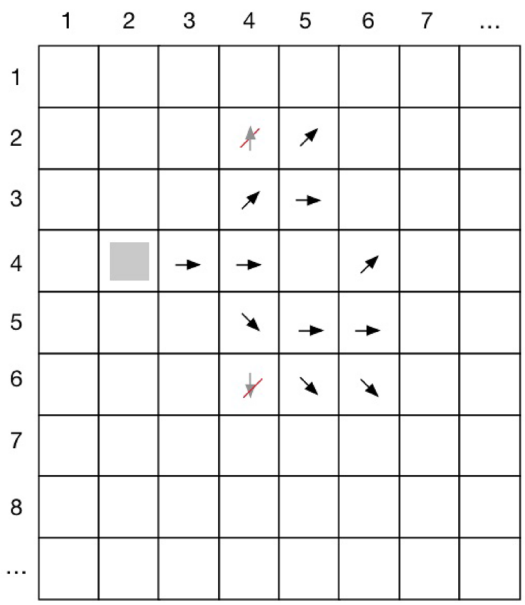

Fig. 7 Example of the adjusted *relay* scheme

message toward the suitable directions. One of the broadcasted messages reaches cell (4, 3) that performs the normal relay. Now, take a look to cell (5, 4). It receives a message that has already performed a clockwise turn, thus such cell does not relay to (6, 4) but only to (5, 5) and (6, 5). Moreover, once cell (5, 4) has relayed the message to cell (5, 5), it has de-facto cancelled the clockwise turn originally performed by such message. Hence, cell (5, 5) can normally relay the message to (6, 6), (5, 6) and (4, 6). This adjusted scheme guarantees that infinite loops do not occur in the mesh also when the offerings are exhausted. Let us note that this additional control, does not affect the correctness of the algorithm. The new *relay* scheme can be realized by means of the *turning* field in the message information, as previously described in the definition of the state transition of the cell. Lastly, with respect to the δ function encoding, it is possible to consider the binary string 01 for messages that have not turned, 10 for messages that have turned clockwise and 11 for messages that have turned counter-clockwise.

A privacy preserving solution

In this section we add a privacy layer to the algorithm described in "Implementation details of the Cellular ANTomata algorithm". We consider shoppers' wishlists a valuable information that should be kept private. The privacy layer we add will hide to shops the products in the wishlists, while allowing shoppers to follow the directions (recommendations) where the sought products are sold. These seem two contrasting requirements, but we can satisfy both by resorting to suitable cryptographic primitives. To hide the product ids one could simply encode (i.e., encrypt) them. Notice that, during the shopping experience, at some point, encoding should be compared to suggesting recommendations. We do not use a randomized encoding (i.e., randomized encryption) as private preserving comparison of randomized encoding are more time-expensive than their deterministic counterparts. Moreover, in our scenario, we do not need to recover the

encoded message (i.e., product's id) but just to compare it to other encoded messages. Therefore, resorting to public (private) key encryption scheme supporting equality test whould be overkill. We could simply encode products ids by using a cryptographic hash function H (a mechanism used to guarantee integrity of information assuring that data has not been tampered with). The function H compresses arbitrary size messages to small (fixed bit-length) strings, it cannot be inverted, but anybody can check whether its value, say y, maps a message m by computing $z = H(m)$ and testing whether z equals y. Obviously, this solution provides no privacy at all as we assume that products' ids are publicly known. Therefore, our solution should be based on some secret information, should be deterministic, and should be secure. We could use either symmetric encryption or a hash-based message authentication code (HMAC) defined below. Since, in our setting, we do not need to decrypt the encoded product ids, we can resort to HMAC. Once the products ids have been encoded, the *intelligence* in the mall should compare shopper's whishlist with shops' recommendations. We could simply solve this problem by letting the shopper sending her encoded wishlist (i.e. shopping list) to the nearest shop. The shop, comparing his/her recommendations with the shopper's wishlist (i.e., computing a set intersection), could suggest to the shopper where to move next. This solution leaks some information on the shopper's list (i.e., its length) and on the direction taken by the shopper. We can avoid such a privacy leaking by resorting to a private set intersection protocol defined below. Thus, to sum up, in our privacy preserving algorithm we will use two cryptographic primitives, namely the HMAC and the Private Set Intersection whose definition we briefly recall below. Notice that, in a centralized scenario one could use searchable encryption scheme based on hidden vector encryption (see [3, 4, 6]). We will explore this possibility in a forthcoming work.

Hash-based message authentication code (HMAC) The cryptographic primitive Hash-based Message Authentication Code allows to compute a digest of a message m by for a given key k (i.e., $\mathsf{HMAC}(k, m)$) by cleverly applying a cryptographic hash function H to m using the key k. According to [21] $\mathsf{HMAC}(k, m)$ can be defined as follows:

$$\mathsf{HMAC}(k, m) = \mathsf{H}(k \,||\, \mathsf{opad} \,||\, \mathsf{H}(k \,||\, \mathsf{ipad} \,||\, m)).$$

Assume that the cryptographic hash function H, on input m, outputs a digest of d bytes by iterating a basic compression function on m's blocks of b bytes. Then, the key k is a random string of any length less than b and

$$\mathsf{ipad} = \text{the byte } \mathsf{0x36} \text{ repeated } -\mathsf{b} \text{ times}$$
$$\mathsf{opad} = \text{the byte } \mathsf{0x5C} \text{ repeated } -\mathsf{b} \text{ times}$$

If there are space constraints, as it could be our setting, we can resort to a *truncated* HMAC that is, instead of considering the whole digest, we can use only t bits (say, the first ones). In [21] it is recommended that t be at least half the length of the hash output and not less than 80 (i.e., according the above description $t \geq \max\{80, 4\mathsf{d}\}$).

Private set intersection (PSI) The cryptographic protocol Private Set Intersection involves two interacting parties: Client with input $C = \{c_1, \ldots, c_w\}$ and Server with input

$S = \{s_1, \ldots, s_v\}$. At the end of the interaction, Client learns $C \cap S$ while Serverlearns nothing. Using traditional secure two-party computation definitions (see [15]), and assuming wlog that $|A| = v$ and $|B| = w$, the PSI functionality can be described as the secure implementation of: $\mathcal{F}_{\mathsf{PSI}} : ((C, w), (S, v)) \mapsto (C \cap S, \perp)$. Previous notation means that Client, with input C and w (i.e., the size of the set held by Server) interacting with Server, with input S and v (i.e., the size of the set held by Client), computes $C \cap S$ while Serverlearns nothing (denoted by $\perp$). We refer to [2, 5, 7] for the description of some PSI protocols.

Privacy preserving shopping: setting and protocol

In this section we show how to add a privacy layer to the algorithm described in "Implementation details of the Cellular ANTomata algorithm". We add such a privacy layer by simply modifying how the shops' *recommendations* are computed and by representing the sets of products available in shops in a different way. Recall that the messages representing the combination of I-HAVE-PRODUCT and RELAY directives, as well as, the list of products available in a shop and the shoppers' whishlist were represented by n-bit binary vectors (i.e., we represented sets by the corresponding characteristic vectors). In our privacy-preserving setting, instead of representing a set through its characteristic vector, we will represent it *traditionally*-as a collection of elements. The boolean operations executed by the algorithm to combine I-HAVE-PRODUCT and RELAY directives into one single message are substituted by the corresponding set operations (i.e., the *or* operation becomes the union, while the *and* becomes the intersection). Overall, the algorithm's structure remains unchanged.

Before describing how to add the privacy layer, we need to set up our notation. In this *new* scenario, we assume that products available in the shopping mall, even though they are sold by different shops, can by identified by a unique alphanumeric reference (e.g., id). In other words, we assume that products' ids are independent of the shops they are sold. Therefore, in our privacy setting, the set $\mathtt{P}$ contains all products' ids sold in the mall. Moreover, we assume that the set of products that are recommended to the shoppers are the ones sold at minimum price within the mall or, more generally, the products sold at minimum price within the products that shops propose as special offers, promotions, or discounts. We denote such a set by $\mathtt{MP}$, this set corresponds to set of available products, referred to as $\mathtt{AP}$, in "Implementation details of the Cellular ANTomata algorithm". The Central Authority (i.e., the Mall Manager) collects the products' prices from all shops, establishes in which shop any given product is sold at minimum price, and define $\mathtt{MP}_c \subseteq \mathtt{MP}$ as the set of minimum-price products available at shop $c \in \mathcal{S}$. We will show later how the Central Authority computes the sets $\mathtt{MP}$ and $\mathtt{MP}_c$ for any $c \in \mathcal{S}$. Now, we can formally describe how the previously introduced sets are defined.

- For any $c \in \mathcal{S}$ we denote by $\mathtt{P}_c \subseteq \mathtt{P}$ the set of products' ids owned by c and by $\mathtt{PL}_c$ the price-list of all products sold by shop c. Namely,

  $\mathtt{PL}_c = \{\langle \mathsf{id}, \mathsf{price} \rangle \mid \mathsf{id} \in \mathtt{P}_c\}$.

  More generally, $\mathtt{PL}_c$ could represent the price-list of all products proposed by c as special offers, promotions, or discounts.

- The set $\texttt{MP}$ of minimum-price products' ids is defined as

$$\texttt{MP} = \left\{ \mathsf{id} \in \texttt{P} \mid \langle \mathsf{id}, \mathsf{price} \rangle \in \bigcup_{c \in \mathcal{S}} \texttt{PL}_c \right\}.$$

- For any $\mathsf{id} \in \texttt{MP}$, we denote by $\texttt{MinPrice}_{\mathsf{id}}$ the minimum price of the product identified by id. Namely,

$$\texttt{MinPrice}_{\mathsf{id}} = \min \left\{ \mathsf{price} \mid \langle \mathsf{id}, \mathsf{price} \rangle \in \bigcup_{c \in \mathcal{S}} \texttt{PL}_c \right\}.$$

- For any $c \in \mathcal{S}$, the set $\texttt{MP}_c$ consists of an *encoding* of products' ids sold by shop c at the lowest price among all the shops in the mall. Namely,

$$\texttt{MP}_c = \left\{ \mathsf{HMAC}(k, \mathsf{id}) \mid \langle \mathsf{id}, \mathsf{price} \rangle \in \texttt{PL}_c \wedge \mathsf{price} = \texttt{MinPrice}_{\mathsf{id}} \right\}.$$

- For any $c \in \mathcal{S}$ and $\mathsf{dir} \in \{N, NE, E, SE, S, SW, W, NW\}$, we denote by $\texttt{MP}_c^{\mathsf{dir}}$ the set of encoded minimum prices received from shop c by the neighbor located towards dir. Message $\texttt{MP}_c^{\mathsf{dir}}$ corresponds to message $\mathbf{msg}_{\mathsf{dir}}$ used by algorithm in "Implementation details of the Cellular ANTomata algorithm" within the transition function δ (see, (7) in "Transition function"). Later, we will show how such a set is computed without privacy loss.
- For any shopper u, her wishlist $\texttt{WL}_u$ contains the ids of the products she is interested in buying.

The privacy preserving protocol is composed of three phases:

- A daily *Initialization Phase* runs by the Central Authority to establish which shop sell which product at the minimum price.
- A *Setup Phase* runs by each shopper entering the mall to *encode* her shopping list.
- A *Shopping Phase* run by the shops to suggest recommendations to shoppers and by any shopper in the mall to decide where direction take for buying the products in her shopping list.

The *Initialization Phase* goes as follows:

- The Central Authority, at the beginning of each day, randomly generates a daily key k to be used for computing the HMAC of products' ids.
- The Central Authority collects the prices of all products sold by the shops in the mall by receiving, from each shop $c \in \mathcal{S}$, the set $\texttt{PL}_c$.
- For each product id, the Central Authority: determines in which shop, say shop c, it is sold at the minimum price; computes the value $\mathsf{HMAC}(k, \mathsf{id})$ and adds it to the set $\texttt{MP}_c$; sends the set $\texttt{MP}_c$ to shop c.

Due to the *HMAC* security, any shop c, not knowing key k, cannot determine which products *belongs* to the sets $\texttt{MP}_c$ (i.e, any shop does not know, from $\texttt{MP}_c$, which products is selling at the minimum price). Shop c can get some information only if either

$|\mathtt{MP}_c| = |\mathtt{PL}_c|$ (i.e., c will learn that he is the cheapest shop in the mall with respect to products appearing in $\mathtt{PL}_c$) or $|\mathtt{MP}_c| = 0$ (i.e., the shop will learn that other shops in the mall sell the products in $\mathtt{PL}_c$ at a lower price). Notice that $\mathtt{MP}_c$ corresponds to q_C in (1) and (2).

The *Setup Phase* goes as follows:

- Any shopper u entering the mall receives from the Central Authority the daily *HMAC* key k.
- The shopper computes the *encoded* version of her shopping list $\mathtt{WL}_u$ by computing the *HMAC* under key k of the products' ids she is interested to. User u stores, into two different sets, the computed *HMAC*s and the tuples $(\mathsf{id}, \mathsf{HMAC}(k, \mathsf{id}))$. More formally, shopper u computes the following sets:

$$\begin{aligned}\mathsf{EWL}_u^{\mathtt{ID}} &= \{\langle \mathsf{id}, \mathsf{HMAC}(k, \mathsf{id})\rangle \mid \mathsf{id} \in \mathtt{WL}_u\},\\ \mathsf{EWL}_u &= \{\mathsf{HMAC}(k, \mathsf{id}) \mid \mathsf{id} \in \mathtt{WL}_u)\}.\end{aligned}$$

At this point, we just need to show how the basic operation of the algorithm in "Implementation details of the Cellular ANTomata algorithm" should be modified in order to have a privacy-preserving shopping experience. The transition function is modeled as follows. Shop c receiving $\mathtt{MP}_c^{\mathsf{dir}}$, for all $\mathsf{dir} \in \{N, NE, E, SE, S, SW, W, NW\}$, computes the messages to be sent to its neighbors by executing similar steps of algorithm in "Implementation details of the Cellular ANTomata algorithm", where boolean operations to merge I-HAVE-PRODUCT and RELAY directives into one single message are substituted by the corresponding set operations (namely, substituting in (9) characteristic vectors with sets and $\vee$ with $\cup$). For instance, the message that will be sent to neighbor located towards E is computed as

$$\mathtt{MP}_c \cup \mathtt{MP}_c^{NW} \cup \mathtt{MP}_c^{W} \cup \mathtt{MP}_c^{SW}.$$

Now, we can describe how the shopper u interacts with shops during her shopping. A simple solution would be for the shopper u to send to shop c her list EWL_u. Shop c computes the following set intersections

$$\begin{aligned}\mathsf{Items}_c &= \mathtt{MP}_c \cap \mathsf{EWL}_u\\ \mathsf{Items}_{\mathsf{dir}} &= \mathtt{MP}_c^{\mathsf{dir}} \cap \mathsf{EWL}_u,\end{aligned}$$

for $\mathsf{dir} \in \{N, NE, E, SE, S, SW, W, NW\}$. Then, shop c sends back the results to shopper u that can make her choice (i.e., where to buy the products she is looking for). The sets Items_c and $\mathsf{Items}_{\mathsf{dir}}$ represent the recommendations suggested by shop c to shopper u. It is clear that such simple protocol leaks some information to shop c as it leaks the directions where there can be found the products the shopper u is interested in and towards she will probably move next. We can avoid such a privacy leak by resorting nine runs of a Private Set Intersection protocol. Indeed, shopper u, approaching a shop $c \in \mathcal{S}$, engages with c nine runs of a Private Set Intersection protocol (PSI protocol, for short) where she plays the role of Client (i.e., she will learn the intersection), while the shop engages the

protocol as Server (learning nothing at the end of the protocol). In particular, the *Shopping Phase* goes as follows:

- In the first PSI protocol run shopper's input is EWL_u, while shop's input is $\mathtt{MP}_c$. At the end of the run, shopper u will compute $\mathsf{Items} = \mathtt{MP}_c \cap \mathsf{EWL}_u$ (she will not have any other information on $\mathtt{MP}_c$ beside to know the elements in both $\mathtt{MP}_c$ and EWL_u); while, the shop c does not gain any information on the shopper's wishlist beside its length.
- If $\mathsf{Items} \neq \emptyset$, then shopper u can compute the products's ids sold by shop c that she is interested in buying. Indeed, for all $item \in \mathsf{Items}$, she will lookup $item$ in $\mathsf{EWL}_u^{\mathtt{ID}}$ getting the corresponding id (remember that $item$ is of the form $\mathsf{HMAC}(k, \mathsf{id})$, for some $\mathsf{id} \in \mathtt{WL}_u$, while $\mathsf{EWL}_u^{\mathtt{ID}}$ contains pairs of the form $\langle \mathsf{id}, \mathsf{HMAC}(k, \mathsf{id})\rangle$, for some $\mathsf{id} \in \mathtt{WL}_u$, too).
- Next eight PSI protocol runs are needed to let shopper u know shop's recommendations (i.e., which direction to follow). In such runs shopper's input[3] is $\mathsf{EWL}_u \setminus \mathsf{Items}$, while, for $\mathsf{dir} \in \{N, NE, E, SE, S, SW, W, NW\}$, shop's input is $\mathtt{MP}_c^{\mathsf{dir}}$. At the end of each of the eight runs, the shop c, beside shopper's wishlist length, does not gain any other information, while shopper u will compute shop's recommendations represented by $\mathsf{Items}_{\mathsf{dir}} = \mathtt{MP}_c^{\mathsf{dir}} \cap (\mathsf{EWL}_u \setminus \mathsf{Items})$. Then, shopper u, analyzing $\mathsf{Items}_{\mathsf{dir}}$ and following her own policy, will head towards a new direction.

Previous protocol based on the cryptographic primitive *Hash-based Message Authentication Code* and the cryptographic protocol *Private Set Intersection* guarantee shopper's privacy. Anyway, the main issue is that any shopper in the mall know the daily key k. This problem will be tackled in the next section.

Increasing privacy level

As we pointed out at the end of previous section, a major concern with the proposed protocol is that any shopper in the mall knows the daily key k used to compute the HMAC of products' ids. Any malicious shopper can leak such a key to a shop c that can try to infer some information from the messages exchanged during the protocol run. For instance, shop c can check whether an item having product id is sold in some shop lying in direction dir simply checking whether $\mathsf{HMAC}(k, \mathsf{id})$ belongs to $\mathtt{MP}_c^{\mathsf{dir}}$. To avoid such a problem, we resort to the *Deterministic Commutative Encryption* primitive and to *some* interaction between the Central Authority and any shopper entering the mall.

Deterministic commutative encryption (DCE) We can define the deterministic commutative encryption primitive by means of the following four algorithms $(\mathsf{Init}, \mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$.

- $\mathsf{Init}(1^\lambda)$: Given a security parameter λ, procedure Init outputs the public parameters pp.

[3] We are assuming that shopper's policy is to buy, as soon as possible, all products in her shopping list. This means that if $\mathsf{Items} \neq \emptyset$, then all products identified by Items will be bought by shopper u at shop c and removed from her wishlist.

- $\mathsf{KeyGen}(pp)$: On input the public parameters pp, procedure KeyGen outputs the secret key[4] sk of the scheme.
- $\mathsf{Enc}(sk, m)$: Given a message m belonging to the message space and the secret key sk, procedure Enc deterministically generates an encryption c of m under secret key sk.
- $\mathsf{Dec}(sk, c)$: Given a ciphertext c belonging to the ciphertext space and the secret key sk, procedure Dec outputs the decryption of c under the secret key sk.

Any *instantiation* of the deterministic commutative encryption primitive has to satisfy the following two properties:

$$\mathsf{Dec}(sk, \mathsf{Enc}(sk, m) = m)$$

and

$$\mathsf{Enc}(sk, \mathsf{Enc}(sk', m)) = \mathsf{Enc}(sk', \mathsf{Enc}(sk, m)),$$

for any pair of keys sk and sk' in the key-space and any message m in the message-space. In other words, one can correctly decrypt and if we encrypt twice the same message under two different secret keys, the order of the encryption does not matter (i.e., we always get the same ciphertext).

The instantiation of the deterministic commutative encryption primitive we consider in this paper is the scheme described in [1] that is based on the Pohlig-Hellman encryption [26].

Now, we can describe how to solve the issue stemming from the shoppers' knowledge of the daily key. The first step of the *Initialization Phase* described in "Privacy preserving shopping: setting and protocol" changes as the daily key used in the protocol to encode the products' ids is randomly generated using the procedure KeyGen of a DCE scheme. More precisely, given a DCE scheme $(\mathsf{Init}, \mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$, the Central Authority generate her/his daily secret key m_k, first, by running $\mathsf{Init}(1^\lambda)$, to obtain the public parameters pp, then by executing $\mathsf{KeyGen}(pp)$ to get the secret key m_k. The Central Authority follows all other steps in the *Initialization Phase* where, instead of computing $\mathsf{HMAC}(k, \mathsf{id})$ she/he computes $\mathsf{Enc}(m_k, \mathsf{id})$.

The *Setup Phase* changes as well and it goes as follows:

- Any shopper u, entering the mall receives from the Central Authority the public parameters pp and computes her secret key u_k by executing $\mathsf{KeyGen}(pp)$.
- User u computes the *temporary* encoding of her shopping list by computing the following two sets:

  $$\mathsf{TmpEWL}_u^{\mathtt{ID}} = \{\langle \mathsf{id}, \mathsf{Enc}(u_k, \mathsf{id})\rangle \mid \mathsf{id} \in \mathtt{WL}_u\},$$
  $$\mathsf{TmpEWL}_u = \{\mathsf{Enc}(u_k, \mathsf{id}) \mid \mathsf{id} \in \mathtt{WL}_u)\}.$$

  and sends TmpEWL_u to the Central Authority.
- The Central Authority encrypts the set TmpEWL_u under the daily secret key m_k by computing the set

  $$\mathsf{DoubleEnc}_u = \{\langle e, \mathsf{Enc}(m_k, e)\rangle \mid e \in \mathsf{TmpEWL}_u\}$$

[4] We describe the DCE primitive for the secret-key setting, an analogous definition can be given for the public-key one where KeyGen outputs a pair of public and private keys.

and send $\mathsf{DoubleEnc}_u$ to shopper u. Notice that any pair in $\mathsf{DoubleEnc}_u$ is of the form $\langle \mathsf{Enc}(u_k, \mathsf{id}), \mathsf{Enc}(m_k, \mathsf{Enc}(u_k, \mathsf{id}))\rangle$ for some $\mathsf{id} \in \mathtt{WL}_u$.

- User u from $\mathsf{TmpEWL}_u^{\mathtt{ID}}$ and $\mathsf{DoubleEnc}_u$ can compute the following two sets:

$$\mathsf{EWL}_u^{\mathtt{ID}} = \{\langle \mathsf{id}, \mathsf{Enc}(m_k, \mathsf{id})\rangle \mid \mathsf{id} \in \mathtt{WL}_u\},$$
$$\mathsf{EWL}_u = \{\mathsf{Enc}(m_k, \mathsf{id}) \mid \mathsf{id} \in \mathtt{WL}_u)\}.$$

Notice that shopper u can compute the two sets defined in the last step of the above *Setup Phase* as, due to the commutative property of the DCE scheme, one has

$$\mathsf{Dec}(u_k, \mathsf{Enc}(m_k, \mathsf{Enc}(u_k, \mathsf{id}))) = \mathsf{Dec}(u_k, \mathsf{Enc}(u_k, \mathsf{Enc}(m_k, \mathsf{id}))) = \mathsf{Enc}(m_k, \mathsf{id}).$$

The transition function δ as well as the *Shopping Phase* have not to be changed. Resorting to a deterministic commutative encryption scheme avoids the need of distributing the daily secret key to all shoppers in the mall increasing the overall privacy level.

Validation and evaluation

In this section we provide the results of the performance evaluation for the distributed algorithm (offering finding) based on Cellular Automata and, subsequently, the results of an early experimentation of a first prototype of the system.

Distributed algorithm evaluation

In order to evaluate the distributed algorithm we have implemented a software simulator of the Cellular ANTomata, tailored to our scenario. The above mentioned software has been implemented as a two-layer application. The lower layer is the Cellular Automata engine developed in Java. The upper layer is a simple graphical Web-based front-end developed in JavaScript and iioEngine[5] framework. The two layers communicate by means of JSON. The software simulator will be used in order to provide a simple case study to graphically show how the algorithm works.

In particular, we prepared the mall as a square mesh of size 10×10 and configured it by inserting three offerings (the red squares) of the same product type and three shoppers (the black diamonds) who are looking for offerings compatible with the product type. Figure 8 contains screenshots of a subset of the simulation iterations. Messages to be relayed are represented by green points and positioned along the direction they are going to be sent. In particular, Fig. 8a provides the screenshot representing the initial configuration of the square mesh. All the cells possessing offerings are ready to relay `I-HAVE-OFFERING` messages to all their neighbors. At step 2 (see Fig. 8b), messages are relayed for the first time and, at step 3 (see Fig. 8c) the shopper originally positioned at (1, 3) moves toward `west` at (1, 2). At step 5 (see Fig. 8d), the above shopper picks up the offering, thus the cell (1, 1) stops broadcasting messages. Figure 8e reports the screenshot describing the situation at step 29 where the shopper originally (see Fig. 8a) positioned in cell (1, 9) picks up the offering in cell (8, 1). Finally, no more offerings are present in the square mesh, no cell generates and broadcasts `I-HAVE-OFFERING`

[5] http://iioengine.com/.

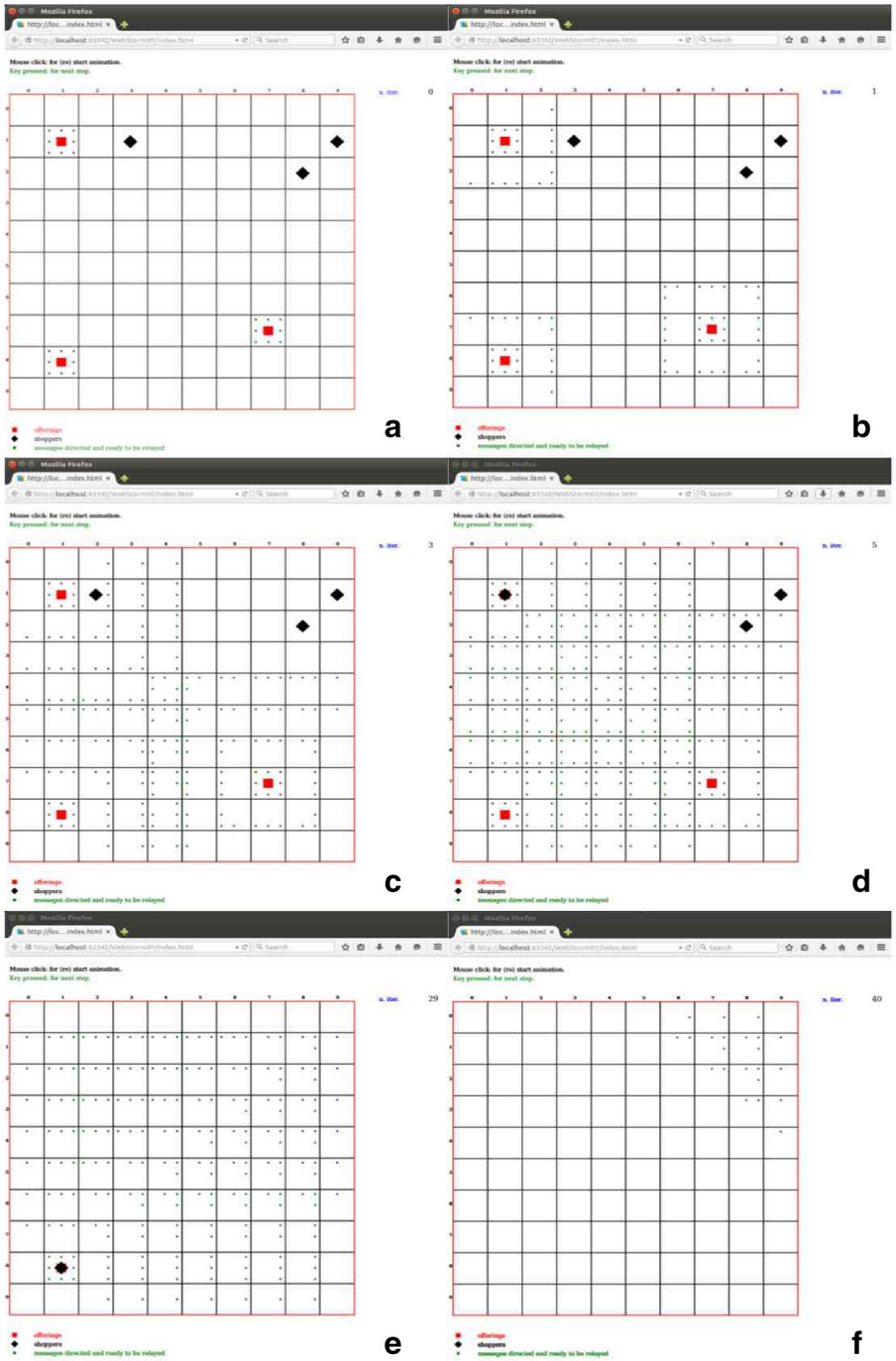

Fig. 8 Simulation at iteration 0, 1, 3, 5, 29 and 40

messages, thus messages in the square tend to disappear as shown in Fig. 8f. The simulation ends at step 45. Actually, in the algorithm, the cell (0, 0) realizes that no other cell possessing offerings is present by means of an FSSP-type synchronization [16].

After the simulation scenario, we evaluated algorithm performances. We used only the lower layer of the simulator over different environment configurations. In particular, the

algorithm has been executed over a dataset of 12 configurations of the same shopping mall that is designed as a square of 20×20 zones. Configuration n.1 includes 2 offerings and 2 shoppers, configuration n.2 includes 3 offerings and 3 shoppers, and so on. Shoppers and offerings are randomly distributed over the square, but configuration i inherits the distribution of configuration $i - 1$. All the configurations are depicted in Fig. 9 where shoppers are depicted in yellow and offerings in red. For the sake of clarity, but without loss of generality, we assume that we have a shopper for each offering and that each shopper has only one item in his/her wishlist that always matches with the offerings, which belong to the same type. We assume also that once a shopper reaches an offering he/she disappears.

Figure 10 shows how many iterations are needed for each configuration to achieve the correct termination of the algorithm (no active available offerings or no active shoppers).

As it is possible to observe in Fig. 10, there is not always a growth of the number of iterations when the configuration increases the number of offerings and shoppers. This is due to the specific distribution of shoppers against offerings. In fact, if you observe configurations n.3 and n.4 you can note that the first one terminates in 42 iterations and the second one terminates in 18 iterations despite the fact that the first one includes a lesser number of offerings and a lesser number of shoppers. Such different performances are due to the new (shorter) distances, among offerings and shoppers, introduced in the configuration n.4. Thus, new shoppers (those introduced in configuration n.4) gather

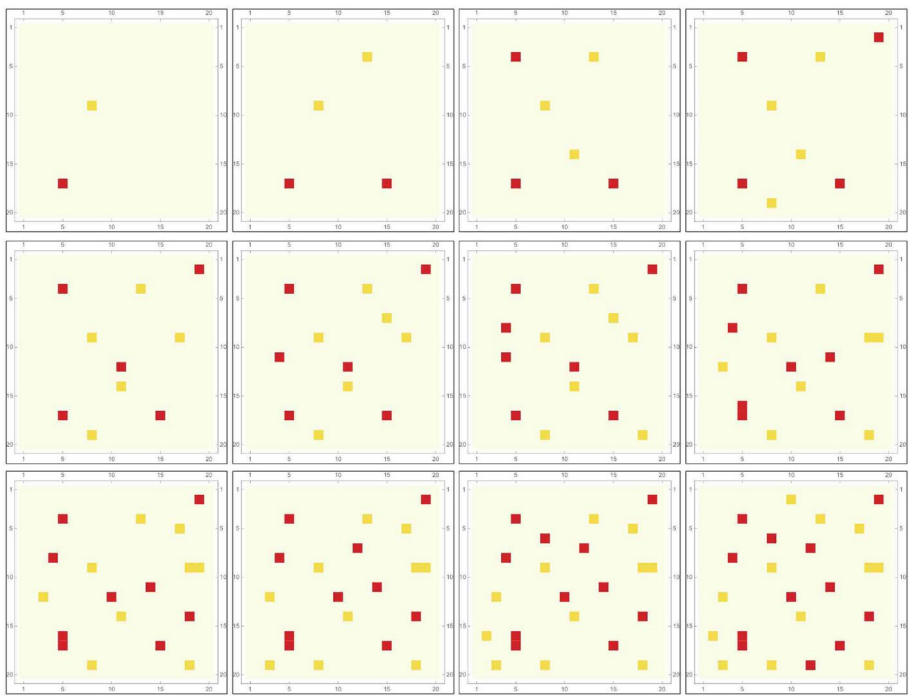

Fig. 9 Configurations of the shopping mall for the second simulation

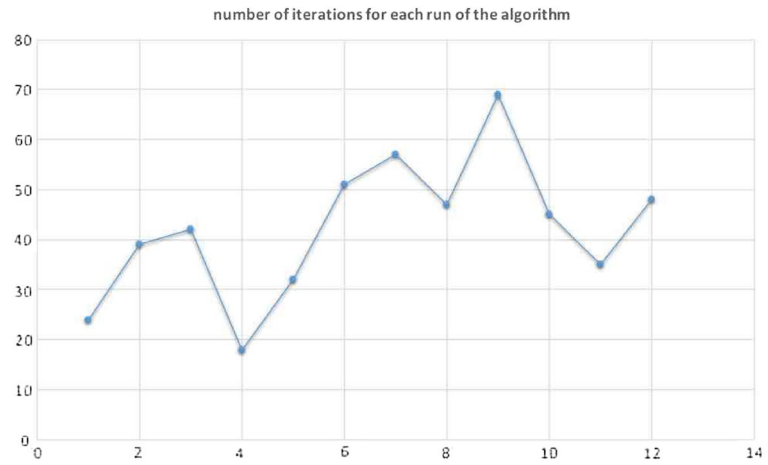

Fig. 10 Number of iterations to achieve the correct termination of the algorithm

first some offerings that, in configuration n.3, require a greater number of iterations because are more distant from the active shoppers.

Early experimentation in a real-world environment

In order to perform an early evaluation of the first prototype of the system we have selected a little shopping mall consisting in 8 shops (with more than one shoes and clothes shops), 1 rest area and 3 corridors. Thus, we have deployed a Raspberry Pi2 for each zone (in total 12 processors) and 12 Bluetooth sensors (one for each processor in each zone). The Central Authority has been deployed by means of a laptop computer. A Wi-Fi LAN has been created to connect each processor to its neighbors and to the Central Authority. In this context, 10 shoppers equipped with smartphones and App to interact with the smart environment are asked to compile their wishlists (composed by 2 items) and follow the recommendations of the system in a given time slice (of 1 h). In this period, the mall provided 20 offerings for products matching at least one of the items in the wishlists of every shoppers.

At the end of the one-hour experimentation, the 10 shoppers were asked to answer the questionnaire reported in Table 1. The questionnaire consisted in 6 Likert scale items, half of them proposed in positive form and the other half proposed in negative form.

Table 1 Questionnaire

Item	Description	Cluster
Q1	The system does not hinder your common behavior within the mall	C
Q2	The system does not improve your shopping experience in the mall with respect to the discovery of interesting offerings	L
Q3	The system correctly suggests directions to reach offerings matching your wishlist or part of it	P
Q4	The system decreases the time for purchasing items in your wishlist or in part of it	L
Q5	The system provides wrong directions for correct offerings with respect to your wishlist or part of it	P
Q6	The system confuses you with respect your purchase intentions in the mall	C

Positive and negative items are randomly distributed. The 5-point Likert Scale, *Strongly Disagree = 1, Disagree = 2, Neutral = 3, Agree = 4, Strongly Agree = 5*, is adopted for all items in the questionnaire. Items have been also grouped in three clusters: *C = Contribute*, *R = Receive*, *P = Perform*. In the first cluster, items are focused on understanding if the recommender system is invasive for the normal behaviors of the shoppers. In the second cluster, items are mostly directed to capture workers' perceptions about usefulness of the recommender system. In the third cluster, items are dedicated to investigate if the recommender system provides correct suggestions to the shoppers. Before starting the analysis, Likert data for negative items are adjusted (for instance, Strongly Agree = 5 becomes 1, Agree = 4 becomes 2, Neutral = 3 remains 3, Disagree = 2 becomes 4 and Strongly Disagree = 1 becomes 5) in order to analyze a coherent dataset. Taking into account the defined clusters of items (i.e. *C*, *L* and *P*), Fig. 11 shows that negative values are present only for the cluster *C* that tries to evaluate the invasiveness of the system. We think that this aspect will be smoothed in the final version of the system, where ergonomic user interfaces and method of interaction with the smart environment will be provided. For cluster *L* that tries to evaluate usefulness, results are positive with only a little percentage of negative answers provided by some shopper. Lastly, results for cluster *P* are very positive. Such cluster tries to evaluate the correctness of the system as perceived by the shoppers.

Conclusion

The main result of the paper is a novel framework to formally model intelligent shopping malls. The result is twofold. It proposes, on one side, a class of application scenarios for cellular automata in the context of ambient intelligence, related to blended shopping. The use of such a formal computation model is a certificate of soundness of the whole framework. On the other side, we provide a protocol to handle privacy in the intelligent shopping ambient with respect to shoppers' information included in their shopping lists. Furthermore, we propose the Offering Finding algorithm for guiding shoppers, within a shopping mall, to reach and pick up offerings for desired products in their shopping lists. A software simulation implementing the above mentioned algorithm has been also developed and described together with an early experimentation case. Future works will provide solutions to guarantee privacy also for merchants, whose instrumented shops

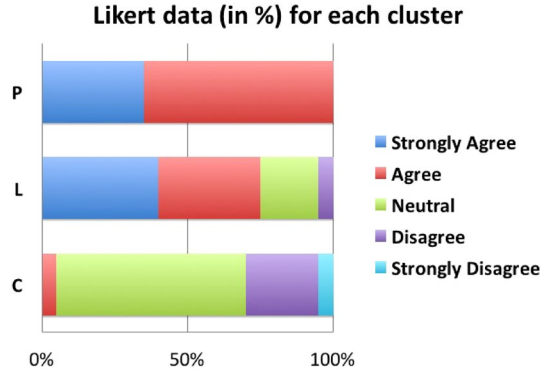

Fig. 11 Likert scale for clusters

exchange messages, providing sensitive offering information, which can be used for unfair competition within the mall. Moreover we plan also to develop an automatic correctness verification, with model checking techniques, of all the presented framework as in [9, 10, 24].

Abbreviations

CA: cellular automata; AmI: ambient intelligence; INS: indor navigation system; FSM: finite state machine; HMAC: hash-based message authentication code; PSI: PrivateSetIntersection; DCE: deterministic commutative encryption.

Authors' contributions

All authors read and approved the final manuscript.

Competing interests

The authors declare that they have no competing interests.

Publisher's Note

Received: 30 May 2017 Accepted: 11 July 2017
Published online: 07 September 2017

References

1. Agrawal R, Evfimievski AV, Srikant R (2003) Information sharing across private databases. In: Proceedings of the 2003 ACM SIGMOD international conference on management of data, pp 86–97
2. Ateniese G, De Cristofaro E, Tsudik G (2011) (if) size matters: size-hiding private set intersection. In: Public Key Cryptography—PKC 2011—14th international conference on practice and theory in public key cryptography, pp 156–173
3. Blundo C, Iovino V, Persiano G (2009) Private-key hidden vector encryption with key confidentiality. In: Garay JA, Miyaji A, Otsuka A (eds) Cryptology and network security: 8th international conference, (CANS 2009), pp 259–277
4. Blundo C, Iovino V, Persiano G (2010) Predicate encryption with partial public keys. In: Heng SH, Wright RN, Goi BM (eds) Cryptology and network security: 9th international conference (CANS 2010), pp 298–313
5. Blundo C, De Cristofaro E, Gasti P (2014) Espresso: efficient privacy-preserving evaluation of sample set similarity. J Comput Secur 22(3):355–381
6. Boneh D, Di Crescenzo G, Ostrovsky R, Persiano G (2004) Public key encryption with keyword search. In: Cachin C, Camenisch JL (eds) Advances in cryptology—EUROCRYPT 2004: international conference on the theory and applications of cryptographic techniques, pp 506–522
7. De Cristofaro E, Tsudik G (2012) Experimenting with fast private set intersection. In: Trust and trustworthy computing—5th international conference, (TRUST 2012), pp 55–73
8. Fallah N, Apostolopoulos I, Bekris K, Folmer E (2013) Indoor human navigation systems: a survey. Interact Comput 25(1):21–33
9. Ferrante A, Napoli M, Parente M (2009a) Graded-ctl: satisfiability and symbolic model checking. Lecture Notes in Computer Science 5885 LNCS, p 306–325. doi:10.1007/978-3-642-10373-5_16
10. Ferrante A, Napoli M, Parente M (2009) Model checking for graded ctl. Fundam Inform 96(3):323–339. doi:10.3233/FI-2009-181
11. Friedewald M, Vildjiounaite E, Punie Y, Wright D (2007) Privacy, identity and security in ambient intelligence: a scenario analysis. Telemat Inform 24(1):15–29
12. Fuchs B, Ritz T, Halbach B, Hartl F (2011) Blended shopping: interactivity and individualization. In: 2011 Proceedings of the international conference on e-business (ICE-B), pp 1–6
13. Gaeta M, Loia V, Orciuoli F, Parmentola M (2013) A genetic approach to plan shopping in the ami-based blended commerce. In: 2013 IEEE international symposium on industrial electronics (ISIE), pp 1–6
14. Gaur MS, Pant B (2015) Trusted and secure clustering in mobile pervasive environment. Hum Centric Comput Inform Sci 5(1):32
15. Goldreich O (1994) Foundations of cryptography. Cambridge University Press, Cambridge
16. Gruska J, La Torre S, Parente M (2004) Optimal time and communication solutions of firing squad synchronization problems on square arrays, toruses and rings. In: Developments in language theory (DLT), Lecture Notes in Computer Science, vol 3340. Springer, Berlin, pp 200–211
17. Gruska J, La Torre S, Napoli M, Parente M (2006) Different time solutions for the firing squad synchronization problem on basic grid networks. ITA 40(2):177–206. doi:10.1051/ita:2006002
18. Gruska J, La Torre S, Parente M (2007) The firing squad synchronization problem on squares, toruses and rings. Int J Found Comput Sci 18(3):637–654
19. Ibrahim N, Mohammad M, Alagar V (2013) Publishing and discovering context-dependent services. Hum Centric Comput Inform Sci 3(1):1

20. Kavakli M (2015) A people-centric framework for mobile augmented reality systems (mars) design: archive 4any. Hum Centric Comput Inform Sci 5(1):37
21. Krawczyk H, Bellare M, Canetti R (1997) Hmac: Keyed-hashing for message authentication. In: RFC 2104
22. Lin Z (2013) Indoor location-based recommender system. Ph.D. thesis, University of Toronto, Toronto
23. Moore EF (1962) The firing squad synchronization problem. Sequential machines, selected Papers, pp 213–214
24. Murano A, Napoli M, Parente M (2008) Program complexity in hierarchical module checking. Lecture Notes in Computer Science 5330 LNAI, p 318–332. doi:10.1007/978-3-540-89439-1_23
25. Olugbara OO, Ojo SO, Mphahlele M (2010) Exploiting image content in location-based shopping recommender systems for mobile users. Int J Inf Technol Decision Mak 9(05):759–778
26. Pohlig S, Hellman M (1978) An improved algorithm for computing logarithms over gf(p) and its cryptographic significance. IEEE Trans Inf Theory 24(1):106–110
27. Purohit A, Sun Z, Pan S, Zhang P (2013) Sugartrail: indoor navigation in retail environments without surveys and maps. In: 2013 10th annual IEEE communications society conference on sensor, mesh and Ad Hoc communications and networks (SECON). IEEE, New York, pp 300–308
28. Rosenberg AL (2008) Cellular antomata: food-finding and maze-threading. In: 37th international conference on parallel processing, 2008. ICPP'08. IEEE, New York, pp 528–535
29. Rosenberg AL (2012) Cellular antomata. Adv Complex Syst 15(06):28
30. Rui C, Yi-Bin H, Zhang-Qin H, Jian H (2009) Modeling the ambient intelligence application system: concept, software, data, and network. IEEE Trans Syst Man Cybern Part C Appl Rev 39(3):299–314
31. Sadri F (2011) Ambient intelligence: a survey. ACM Comput Surv 43(4):36
32. Siekkinen M, Hiienkari M, Nurminen JK, Nieminen J (2012) How low energy is bluetooth low energy? Comparative measurements with zigbee/802.15.4. In: 2012 IEEE wireless communications and networking conference workshops (WCNCW). IEEE, New York, pp 232–237
33. Umeo H, Kubo K (2010) A seven-state time-optimum square synchronizer. In: Bandini S, Manzoni S, Umeo H, Vizzari G (eds) Cellular automata, Lecture Notes in Computer Science, vol 6350. Springer, Berlin, pp 219–230
34. Vasilakos A, Pedrycz W (2006) Ambient intelligence, wireless networking and ubiquitous computing. Artech House Inc, Norwood
35. Watson RT (2000) U-commerce: the ultimate. Ubiquity. doi:10.1145/353165.353882
36. Winkler C, Broscheit M, Rukzio E (2011) Navibeam: indoor assistance and navigation for shop-ping malls through projector phones. In: CHI 2011 workshop on mobile and personal projection
37. Yang WS, Cheng HC, Dia JB (2008) A location-aware recommender system for mobile shopping environments. Expert Syst Appl 34(1):437–445