An ACO-based personalized learning technique in support of people with Acquired Brain Injury

Kamil Krynicki[a,*], Javier Jaen[a], Elena Navarro[b]

[a] *ISSI Research Group*
Departamento de Sistemas Informáticos y Computación
Universitat Politècnica de València
Camino de Vera, s/n, 46022 Valencia, Spain
Phone: +34 96 387 35 69, Fax: +34 96 387 73 59
[b] *LoUISE Research Group*
Computing Systems Department
University of Castilla-La Mancha
Avda España s/n, 02071 Albacete, Spain

Abstract

The ever-increasing cases of acquired brain injury (ABI), especially among young people, have prompted a rapid progress in research involving neurological disorders. One important path is the concept of relearning, which attempts to help people regain basic motor and cognitive skills lost due to illness or accident. The goals of relearning are twofold. First, there must exist a way to properly assess the necessities of an affected person, leading to a diagnosis, followed by a recommendation regarding the exercises, tests and tasks to perform; and second, there must be a way to confirm the results obtained from these recommendations in order to fine-tune and personalize the relearning process.

This presents a challenge, as there is a deeply-rooted duality between the personalized and the generalized approach. In this work we propose a personalization algorithm based on the Ant Colony Optimization (ACO), which is a bio-inspired meta-heuristic. As we show, the stochastic nature of ants has certain similarities to the human learning process.

We combine the adaptive and exploratory capabilities of ACO systems to respond to rapidly changing environments and the ubiquitous human factor. Finally, we test the proposed solution extensively in various scenarios, achieving high quality results.

Keywords: ACO, recommendation system, ABI, acquired brain injury, relearning process

*Corresponding author
Email addresses: kkrynicki@dsic.upv.es (Kamil Krynicki), fjaen@dsic.upv.es (Javier Jaen), Elena.Navarro@uclm.es (Elena Navarro)

1. Introduction

Computer-aided recommendation, which is now extensively used, ranges from content recommendation for ad-related issues to music and video suggestions. A particular niche of the subject is the learning unit recommendation, which usually consists of suggesting to a student the next learning unit to complete with e-learning software. The computation techniques behind these problems are plentiful, ranging from fuzzy logic, neural networks to ant colony optimization (ACO). Due to its nature, ACO is very well suited for establishing paths in a discrete space, which can be seen as a metaphor for progressing through a given set of study material. An interesting case is one where the search over the space of learning units is not limited to a sequence, but rather centers on finding the most suitable unit for a given user at a given time.

One particular use of the learning schema is the *relearning process*, which is, as the name suggests, the process of learning abilities that have been lost. Relearning is especially important in the context of people with Acquired Brain Injury (ABI). The World Health Organization states that people with ABI have suffered "Damage to the brain, which occurs after birth and is not related to a congenital or a degenerative disease. These impairments may be temporary or permanent and cause partial or functional disability or psychosocial maladjustment." [1]. The causes of ABI vary; they include, but are not limited to skull-brain trauma, degeneration of the blood vessels, meningitis and brain tumors. ABI affects many people every year; in the United States alone 1.7 million people suffer brain injury each year [2]. The impact of ABI is wide-ranging, from a multitude of physical effects, such as muscle spasticity, paralysis or weakness, to cognitive abilities, such as memory, thinking skills or concentration, and organization or planning abilities.

There is a general consensus that an automatized relearning process has a high impact on the individuals affected by ABI. However, this process must be carefully customized to individual abilities/disabilities, since no two people can expect to have the same outcome or problem, even after a seemingly similar brain injury. Proposing a relearning model is more complex than the straightforward learning unit recommendation or progressing through a sequence of tasks. For people with ABI there is no recognized path through the space of relearning activities to success. Instead the most appropriate activities must be found for each new situation. In addition, the software must take into account the user's cognitive and motor skills to an extent that is simply unnecessary for classical learning software.

In this work we present an ACO-based recommendation technique that incorporates the above assumptions and concerns. This technique suggests activities to individuals with ABI, starting with those that have proven to be the most generally successful, and then progressing towards more personalized recommendations, at all times taking into account the user's mental and physical state.

The remainder of this paper is structured as follows. Section 2 discusses related works and highlights the novelty of the present work. Section 3 presents the problem domain in more detail. Section 4 presents the mathematical definition of our ACO algorithm for personalized learning. Section 5 describes the experiments performed with their results and a discussion. Finally, Section 6 presents our main conclusions and areas of future work.

2. Related Work

E-learning systems are powerful tools that complement or even replace regular teaching/learning activities. As mentioned, the relearning process for individuals with ABI is also a learning/teaching process that they must go through to recover their lost abilities. The reasons for the adoption of e-learning systems are many, but the fact that it is available at all times and places is one of its most attractive aspects. However, despite undeniable advantages, their use can be frustrating for educators, as they can find it difficult and time consuming to define the paths each learner has to follow [3]. For this reason, the introduction of *recommendation systems* [4] meant a meaningful step forward for both, educators and learners. These systems have taken advantage of different techniques, such as Collaborative Filtering [5], Ant Colony Optimization [6], Data mining [7], particle swarm optimization [8], biogeography-based optimization [9] or combinations of different techniques [10].

Unfortunately, most of the recommendation systems simply exploit learner preferences, interests and browsing behaviors as input to recommend learning activities. As stated in [11], these approaches miss an important point, which is taking into account the abilities of the learner, in order to truly equip e-learning systems with *personalized learning mechanisms*. De Maio et al. [12] encourage the exploitation of personalized e-learning as the way to become "more effective and efficient when a great number of educational content requires to be dynamically filtered and assembled with respect to learners' preferences and cognitive states." The user's cognitive state thus becomes the cornerstone of the recommendation systems in the personalization process.

The development of personalized e-learning systems has received a great deal of attention from the Artificial Intelligence community. Some studies have promoted the use of artificial neural networks to implement the aforementioned aspects of intelligence. For instance, Baylari and Montazer [13] developed a multi-agent system able to estimate the learners' abilities in order to provide them with tests personalized and adapted to those abilities. One of the main strengths of this work is the exploitation of an artificial neural network to discover users' learning problems when using the system and then recommend them appropriate learning material. Another interesting proposal is called MASACAD [14], a multi-agent e-learning system able to provide students with academic assistance by using an artificial neural network based on their preferences to infer such advice.

Fuzzy systems have also been used in the development of personalized e-learning systems. Lu et al. [10] applied this approach in a system that provides learners with learning objects by analyzing both, the course difficulty of the material and the abilities of the learner. They employ a fuzzy set clustering algorithm to create clusters of similar learners and assign them learning objects.

The introduction of evolutionary algorithms has also been exploited in the context of personalized e-learning systems. For instance, Maio et al. [12] developed a system, named Intelligent Web Teacher (IWT). IWT builds the user's learning context (current needs, cognitive abilities and preferences) in an automatized way, personalizing the learning experience through ad-hoc educational paths. One of the main foundations of IWT is that the generation of learning paths for a specific learner, or a class of learners, is transformed into an optimization problem. Authors combine global and local searches to perform the exploration of the set of learning objects to find the most appropriate learning path that includes the target concepts. Another interesting work is that presented by Acampora et al. [15], who propose the definition of sequences of learning objects in terms of competencies and transform the problem of finding the proper sequence of learning objects into a classical Constraint Satisfaction Problem (CSP). They chose, however, to use a particle swarm optimization algorithm in their solution.

Other interesting proposals have also been developed by employing ACO algorithms. Sharma et al. present in [16] an ACO based algorithm, named Adaptive Content Sequencing in eLearning (ACSeL), that uses ant colonies to evaluate the learning paths and the learners' profiles to recommend suitable content. It is worth noting the dynamic nature of the proposal, as it takes into account the varying (most likely increasing) knowledge levels of the learners in order to tune the strategy to produce a recommendation. Some other studies [17] [18] [19] also used an ACO model to analyze past learning experiences to discover new learning paths. Although most proposals use previous learning experiences to update the pheromone trails as presented by ACSeL, they tend to consider the best solution only, in order to avoid long convergence times. An interesting aspect here is the introduction of a training strategy that promotes the division of learning goals into sub-goals to fit the time available for the learning session.

In a similar way to some of the aforementioned authors, in our proposal we opted for an evolutionary algorithm (ACO), which has been deemed a very promising underlying technology. Our contributions to the basic setup are multiple. First, an important feature of our solution is that it takes into account the state of the individuals with ABI to drive the optimization search. Second, a remarkable property of our system is that it is dynamic and able to adapt the recommendation according to the recent behavior of, not only the user in question, but an entire cluster of users of similar characteristics. Third, it also takes into consideration the cognitive state of the user as the set of deficits (deductive reasoning, sustained attention, short-term memory, etc.), which is later treated as a characteristic unrelated to the knowledge levels. In addition, the most important non-technical contribution is the fact that it is the first use of evolutionary techniques within the domain of ABI, and as such, it is the only fully distributed, decentralized and abstract ABI recommender system. This means that any expert can contribute his preferred set of relearning tasks to the network. Such new units are discovered, presented to the users and, from that point on, they undergo continuous evaluation, with a chance of substituting the current *best-discovered* units, for some particular cases. A growing network of relearning tasks would allow a more precise matching user-deficiency-task, with potentially pivotal benefits. Finally, from a technical point of view, our solution does not focus on actually finding a learning path, as in all the cited proposals, but on the selection of a ranked set of learning activities. To the best of the authors' knowledge no similar proposals have previously been defined for people with ABI.

3. Problem Domain

ABI, as defined in the Introduction, may result from a number of different causes, either internal or external. Internal causes are the most frequent in the elderly, usually due to vascular disorders, such as strokes or hemorrhages. External causes, generally known as traumatic brain injury (TBI), are usually due to traffic accidents, falls, etc. As it can be seen from these examples, every one of us is exposed to this problem at any point in our lives. This explains why the number of people with ABI is growing every year and is currently one of the most frequent health problems. For instance, according to the Brain Injury Center [20] TBI is more common than breast cancer, spinal cord injury, HIV/AIDS, and multiple sclerosis (MS) combined. In the United Kingdom alone it is estimated that at least 1 million people live with the long-term effects of brain injury [21]. In fact, ABI is recognized throughout the world as a problem of epidemic proportions, known as "the Silent Epidemic".

Brain damage can result in different long-term deficits, depending on the area injured and the level of damage. These deficits can be classified into four categories: i) physical deficits that limit the control of a part of the body, such as paralysis or motor coordination; ii) cognitive deficits that impair intellectual performance, such as memory problems; iii) emotional problems that limit or change the control of the feelings, such as depression or anxiety; iv) behavioral deficits that negatively affect the interaction with the environment, such as irritability and restlessness. Although physical deficits are difficult for people to adapt to [22], the cognitive ones are highly disabling, as they interfere with the rehabilitation process and have the most negative effect on the quality of life [14].

The following impairment types are usually related to cognitive deficits [23]:

- Executive function impairments: problems in controlling and regulating activities or behaviors that include abstraction, categorization (the ability to recognize objects and actions), cognitive flexibility (the ability to adapt cognitive processing to new and unexpected situations), deductive reasoning, planning and problem solving.

- Attention impairments. These can be detected when a person with ABI exhibits problems with abilities [24] such as sustained attention (to direct and focus a cognitive activity, given a specific series of stimuli), divided attention (to be able to simultaneously respond to multiple stimuli), or selective attention (to be able to identify the relevant stimulus while several distracting stimuli are generated).

- Memory impairments: including short-term memory, semantic memory, related to the ability to collect information and knowledge about the world without considering previous experiences, episodic memory of personal events, such as places and emotions, which can be explicitly stated and procedural memory, based on implicit learning, mainly for motor skills.
- Language deficits: detected when a person has difficulties in understanding or communicating, reading or understanding a document.
- Spatial perception deficits: people with these deficits exhibit difficulties with construction activities that require spatial abilities.

There is increasing evidence that individuals with ABI should be provided with proper treatment as soon as possible [25]. This treatment is often carried out in a center where they perform a number of supervised activities, usually employing a board game or learning cards. However, this alternative has several drawbacks, especially in terms of the time available for the relearning process, since it is highly dependent on the number of specialists available. In addition, Christiansen et al. [26] confirmed that the use of computers in the relearning process helps to encourage and stimulate cognitive behavior and allows the disabled to reinstate damaged functions.

The concept of using e-learning systems for their treatment thus emerges in a natural way. One of these systems is called HABITAT [27, 28] and has been developed by the authors of this work in collaboration with the ABI Association of Castilla-La Mancha (ADACE). For its development, researchers from the University of Castilla-La Mancha carried out continuous tracking of the relearning process of the handicapped over a period of two years. This study was designed to determine how the relearning process could be supported by software, what different kinds of relearning activities should be provided by an e-learning system and how they could be created and parameterized by specialists.

All this experience was documented as a catalog of relearning activity patterns (ReAP) [29]. Twenty-three cognitive activity patterns were identified and their names are shown graphically in Fig. 1, each corresponding to different cognitive deficits that people with ABI can suffer. Although each one of these patterns treats several cognitive deficits, they have been defined to focus on a special cognitive area. This explains why they have been related in Fig 1. For instance, it can be observed that Question Evocation and Visual are related because they treat mainly memory impairments, such as short-term memory and semantic memory. Within the scope of the activity patterns specialists create new relearning activities, such as the one illustrated in Fig. 2, customized according to the specific needs of the individual subjects.

One of the most demanding tasks that specialists have to carry out is the design of the relearning process of each individual. They have to select learning

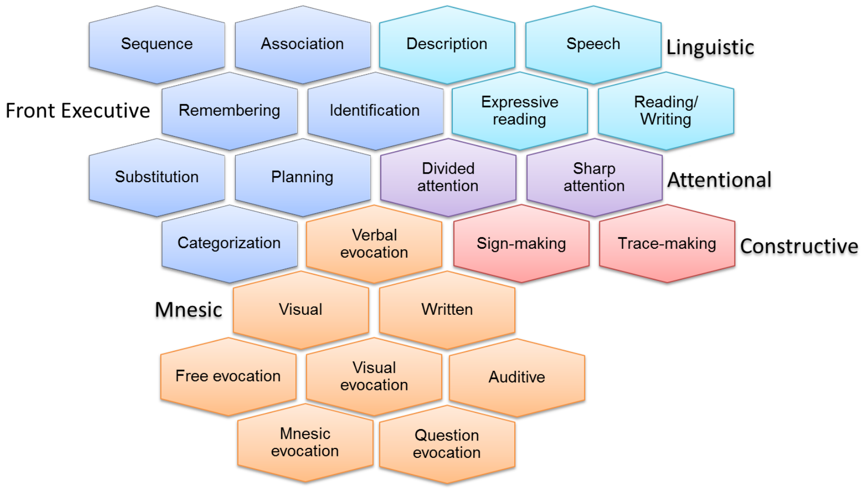
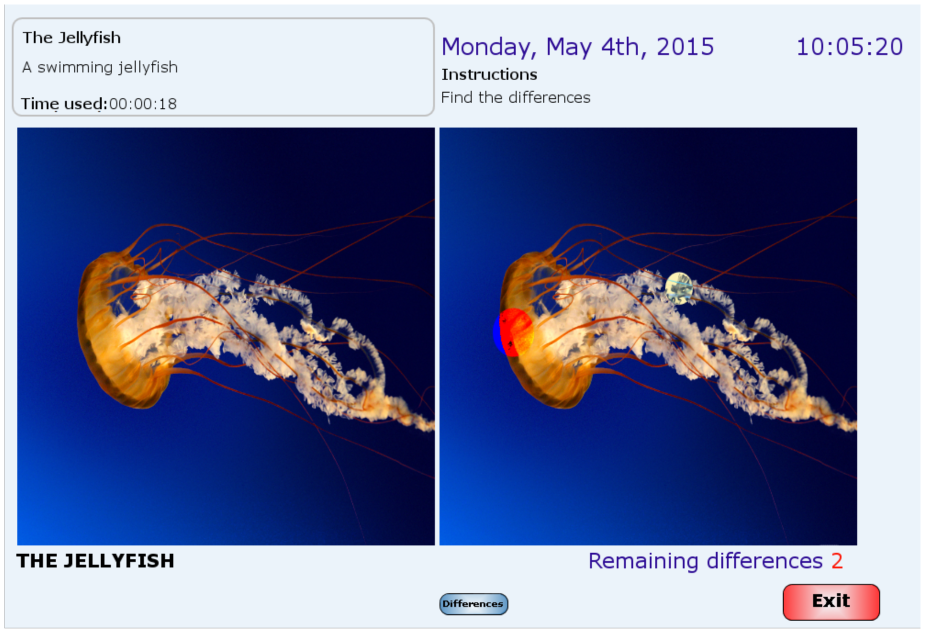


Figure 1: ReAP catalog with 23 cognitive activity patterns.


The Jellyfish

A swimming jellyfish

Time used:00:00:18

Monday, May 4th, 2015 10:05:20

Instructions

Find the differences

THE JELLYFISH

Remaining differences 2

Figure 2: Divided Attention Relearning Activity.

activities taking into account a range of different characteristics, such as age, damage level, stress, etc. of the person being treated. Due to the success of the e-learning software, we can claim that this step can be largely simplified or even eliminated by the proper use of dynamic e-learning models. The definition and development of such a recommendation technique would be a valuable asset for both specialists and people affected by ABI alike. The former could devote more time to the treatment and the latter would benefit from a relearning process specially adapted to their needs.

4. An ACO Algorithm for ABI Rehabilitation Tests Recommendation

4.1. Ant Colony Optimization

ACO [30] is a non-deterministic evolutionary algorithm based on the use of simple and stochastic automata to perform complex optimization tasks. The automata in question are like ants in a colony searching for food and resources. The mathematical model behind ACO is simple, yet powerful; each ant may find itself in two states - either searching for food randomly, the so-called exploration phase, or following the established paths, the exploitation phase. Without the loss of generality, the worlds in which the ants live are limited to bidirectional graphs rather than continuous open spaces.

Every search for resources must begin and conclude in one of the nodes of the system. Once an ant is brought to life, it is given an objective and end conditions; as soon as they are fulfilled the ant returns to the emitting node and reports its findings. In each node it visits, it collects resources corresponding to the goal associated. Pheromones are the essence of inter-ant communication; ants either follow or deposit the pheromone according to the quality of their findings and the mode of behavior they choose.

In short, the basic components of the ACO model are: i) graphs, composed of nodes; ii) links between nodes, with assigned values of pheromone and cost; iii) the resources in each node, representing goods of various types; iv) ants, the search automata. Each of these elements finds its corresponding metaphor in our conceptual model, as explained in Subsection 4.2.

4.2. Problem space conceptual model

In Fig. 3 we present the conceptual model of the problem space. The two main components are: i) *user*, which represents a person, with its basic data and ii) *test pattern*, or a *relearning activity* pattern p in the ABI domain (see Section 3), which is an abstraction of a group of tests t. An example of a test pattern would be the *Association* test pattern, under which specialists create tests containing a particularized set of images and texts to match, depending on the specific abilities to treat.

Each user u is described by i) an impairment state e_u which is denoted A, B, C, D or E. From A to D the degree of cognitive impairment increases, and E indicates a person in coma who does not receive any treatment as explained in [29]; ii) the list of acquired injuries $dc_u = [dc_1, dc_2, dc_3, \ldots]$, in order of

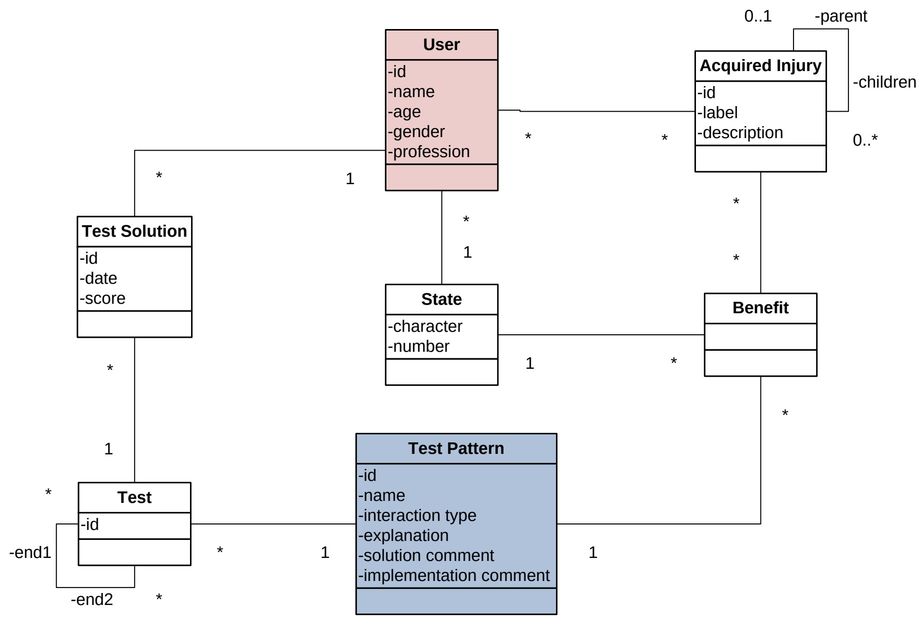


Figure 3: Conceptual Model.

importance. The first acquired injury is referred to as main acquired injury; iii) additional metadata that are taken into account, such as the user's age w_u (integer value) and gender g_u (boolean value, 1 for female, 0 for male), as well as the profession j_u, drawn from the hierarchical structure provided by [31]. The full description of a user is written as: $u = (e_u, dc_u, j_u, w_u, g_u)$.

On the other hand, each test pattern p provides a set of benefits dc_p for users in given states e_i and acquired injuries dc_i, written as: $dc_p = \{(e_1, \{dc_{11}, dc_{12}, \dots\}), (e_2, \{dc_{21}, dc_{22}, \dots\}), (e_3, \{\dots\}), \dots\}$. The dc_p notation can be read as: for users in the state e_1 the test will be beneficial in case of the acquired injuries $dc_1, dc_2, \dots$, etc. Note that each state corresponds to a set of acquired injuries rather than a list, therefore the order of acquired injuries does not reflect the increase or decrease of efficiency of the test pattern. In addition, not all possible states must be benefited by a given p. For instance, state E is never benefited, as it is the comatose state, which means user-system interaction is impossible. All the test patterns available in the system are organized in a tree-like structure that reflects dependencies and relates them to each other.

We argue that this approach, alongside underlying ACO strategy, is sufficient to model the user–system interactions. The key algorithmic problem is to propose a measure that would be capable of reflecting the satisfaction level of a user u with a test pattern p that goes beyond a straightforward scoring system, which is given with ACO. ACO performs a simple goodness marking, which is a very good base to build upon, but it can be greatly improved by incorporating more features of our particular problem-space. Our proposal bases the measure on the performance history of all the users, as well as the potential match between the list of user's acquired injuries and test pattern's benefits. We elaborate on the subject in detail in Subsection 4.3.

The modeling of the problem space with the concepts offered by the ACO domain is as follows: i) tests t are represented by nodes N; ii) tests t are organized in a graph structure K, which is the aggregate of all the tests available; iii) the action of querying tests for a user u is implemented as a release of a number of virtual ACO ants onto K; iv) the action of solving tests produces a test solution Ts object, which is related the test in question t and the user involved u, as well as the completion date d and score s and written as $Ts = (t, u, d, s)$; v) test solutions Ts are the resources of the model. They are evaluated and collected by ants and stored in nodes N of K.

The graph K has a toroidal topology with random distribution. In the work [32] we demonstrate that the topology has only a minor impact on the efficiency. We therefore decided not to take this property as a factor in our study.

As mentioned, each ant models the action of searching for a test pattern suitable for a given user. The ants are routed following a well-known ACO algorithm, the Ant Colony System (ACS) [33]. For each visited test in the node n_i the ant generates an evaluation response $res_i = (n_i, s_i)$, where s_i is the score of the node for the given query. See Algorithm 6 for details on how the responses res_i are obtained. The algorithmic mechanics behind the process of querying is explained in Subsection 4.3.

It is important to point out here our approach to the pheromone concept. Traditionally, there is one layer of pheromone on top of the underlying graph. This means that each ant reads and writes the same values. In some studies, however, various levels of pheromone per graph have been introduced [32, 34]. This matches our situation, as in our problem-space there are many diverse users with unrelated disabilities. Therefore, we use one pheromone level per disability. As a consequence, the ants are introduced only to the level corresponding to the main disability of the user related to the query.

4.3. Algorithmic Design

Our algorithmic design allows users to query for tests at any moment, requiring as input the minimum and a maximum normalized score of tests to solve, s_{min} and s_{max} respectively. The higher the s_{min} the higher the match quality required to present the test found to the user. The lower the s_{max} the more diverse and unexpected the suggestions become. The narrower the gap between the s_{min} and the s_{max} the smaller the number of results becomes. All the tests that have been discovered to have the similarity $s \in [s_{min}, s_{max}]$ will be presented to the user in descending order by the s value.

High level recommendation querying is governed by Algorithm 1. Once the query is launched, the system releases a series of ants in search of tests. During the querying process, the ants evaluate nodes along the way. The ants are routed according to the traditional ACS rules, as mentioned in Section 4.1, with the exception of the pheromone, which has been divided into layers. Each pheromone layer corresponds to an acquired injury dc.

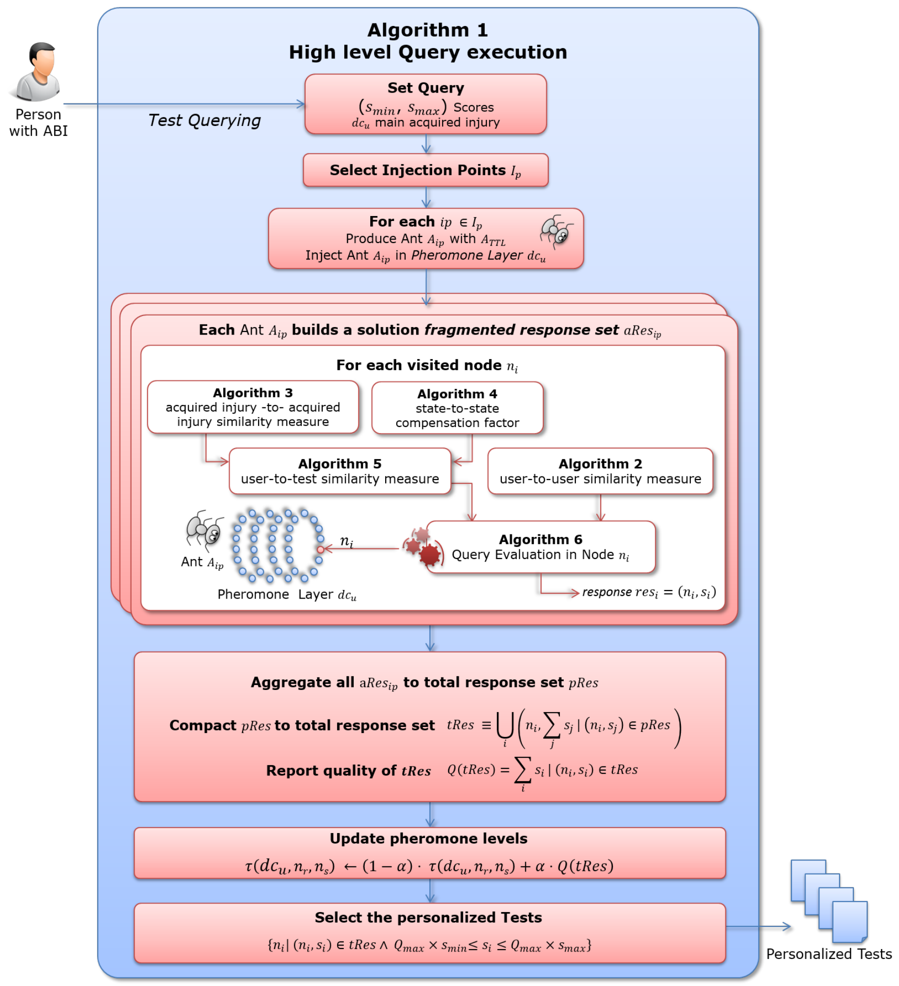

Algorithm 1
High level Query execution
Person with ABI
Test Querying
Set Query
(s_{min}, s_{max}) Scores
dc_u main acquired injury
Select Injection Points I_p
For each $ip \in I_p$
Produce Ant A_{ip} with A_{TTL}
Inject Ant A_{ip} in Pheromone Layer dc_u
Each Ant A_{ip} builds a solution fragmented response set $aRes_{ip}$
For each visited node n_i
Algorithm 3 acquired injury -to- acquired injury similarity measure
Algorithm 4 state-to-state compensation factor
Algorithm 5 user-to-test similarity measure
Algorithm 2 user-to-user similarity measure
Ant A_{ip}
n_i
Algorithm 6 Query Evaluation in Node n_i
Pheromone Layer dc_u
response $res_i = (n_i, s_i)$
Aggregate all $aRes_{ip}$ to total response set $pRes$
Compact $pRes$ to total response set $tRes \equiv \bigcup_i \left(n_i, \sum_j s_j \mid (n_i, s_j) \in pRes\right)$
Report quality of $tRes$ $Q(tRes) = \sum_i s_i \mid (n_i, s_i) \in tRes$
Update pheromone levels
$\tau(dc_u, n_r, n_s) \leftarrow (1-\alpha) \cdot \tau(dc_u, n_r, n_s) + \alpha \cdot Q(tRes)$
Select the personalized Tests
$\{n_i \mid (n_i, s_i) \in tRes \wedge Q_{max} \times s_{min} \leq s_i \leq Q_{max} \times s_{max}\}$
Personalized Tests

Figure 4: Graphical description of Algorithm 1.

Algorithm 1: High level query execution

```
1: assume
     - user $u$ with main acquired injury $dc_u$
     - minimum $s_{min} \in \mathbb{R}_0^1$ and maximum $s_{max} \in \mathbb{R}_0^1$ desired score
     - a subset of the nodes of the graph $K$, called injection points $I_p$
2: for each injection point $i_p \in I_p$ produce an ant $A_{ip}$ and release it into the
   pheromone layer corresponding to $dc_u$
3: while all $A_{ip}$ not finished do in parallel $A_{ip}$ builds a proposal of a solution
   $aRes_{ip}$ (ant response set): a response $res_i = (n_i, s_i)$ for every node $n_i$ visited
```

```
4: **end while**
5: **let** $pRes$ (partial response set) be the aggregate of the $aRes_{ip}$. $pRes$ contains
   repetitions with respect to the $n_i$ value, as various ants might have evaluated
   the same node.
6: **obtain** total response set $tRes$, by removing the repetitions from the partial
   response set as shown in (1)
```

$$tRes = \bigcup_i \left(n_i, \sum_j s_j | (n_i, s_j) \in pRes \right) \tag{1}$$

```
7: **obtain** the total (2) and top (3) quality of $tRes$:
```

$$Q(tRes) = \sum_i s_i | (n_i, s_i) \in tRes \tag{2}$$

$$Q_{max}(tRes) = \arg\max_s (n_i, s_i) \in tRes \tag{3}$$

```
8: Update pheromone values on links that participated in the solution:
```

$$\tau(dc_u, n_r, n_s) \leftarrow (1 - \alpha) \cdot \tau(dc_u, n_r, n_s) + \alpha \cdot Q(tRes) \tag{4}$$

```
   where:
   - $\tau(dc_u, n_r, n_s)$ is the pheromone value, in the pheromone layer $dc_u$, between
     the tests (nodes) $n_r$ and $n_s$
   - $\alpha \in \mathbb{R}_0^1$ regulates the influence of the new pheromone
9: The final result consists of tests $t$ associated with nodes $n_i$ that fulfill (5)
```

$$\{n_i | (n_i, s_i) \in tRes \land Q_{max} \cdot s_{min} \leq s_i \leq Q_{max} \cdot s_{max}\} \tag{5}$$

Our algorithmic design, as described, is based on the definition of similarity functions between users (Algorithm 2), between acquired injuries (Algorithm 3) and between user states (Algorithm 4). Fig. 4 offers a graphical description of Algorithm 1 to illustrate how the different subsequent algorithms are used.

The ability to compare different users is essential. It enables us to extrapolate the behavior of one user to recommend tests to users without performance history. In (6) we obtain a user-to-user distance $d(u_1, u_2)$ as a weighted linear combination of several subdistances that have been deemed the most relevant. Apart from the natural distances of disability and physical state we chose to distinguish users of different professional backgrounds. The gender- and age-distribution is based on the fact that the same disabilities acquired at different moments and situations in life can have different sources and, therefore, must not be considered completely identical. With (7), we obtain a smooth and normalized similarity function sim_{u-u} with the softest and most gradual curve around the mean of the population. The function used for the transformation of the unbound distance value into a bound similarity level must have the following properties: i) domain of $\mathbb{R}$; ii) upper- and lower-bound codomain, limits approached asymptotically; iii) be antisymmetric and monotonically growing. We chose arctan, as it fulfills the above properties. We transformed arctan to center on the $(0,0)$ point and normalized it with π^{-1} factor.

Algorithm 2: sim_{u-u}, user-to-user similarity measure

```
1: assume
   - users $u_1 = (e_1, dc_1, j_1, w_1, g_1)$ and $u_2 = (e_2, dc_2, j_2, w_2, g_2)$, where: $e_i$ is the physical state, $dc_i$ is the main acquired injury, $j_i$ is the profession, $w_i$ is the age and $g_i$ is the gender of the $i$-th user, for $i = 1$ and $i = 2$.
2: obtain
   $d(u_1, u_2) = 1 + \alpha_{dc} \times d(dc_1, dc_2) + \alpha_j \times d(j_1, j_2) + \alpha_w |w_1 - w_2| + \alpha_g |g_1 - g_2|$   (6)
   where
   - $d(dc_1, dc_2)$ and $d(w_1, w_2)$ are the shortest distances in the corresponding hierarchical structure between the values in question
   - the parameter values: $\alpha_{dc} = 0.7$, $\alpha_j = 0.1$, $\alpha_w = 0.1$, $\alpha_g = 0.1$, and serve as weights in the equation. The most important component was given a dominating value, the rest is evenly distributed among the remaining factors.
3: obtain
   $sim_{u-u}(u_1, u_2) = \frac{1}{2} - \frac{1}{\pi} \times \arctan\left(\frac{d(u_1, u_2) - \mu_d}{\sigma_d}\right)$   (7)
   where
   - $\mu_d$ and $\sigma_d$ are the mean and the standard deviation of the user population
```

Algorithm 3 calculates the similarity between acquired injuries, sim_{d-d}. As the acquired injuries are provided in a hierarchical structure [29] we derive the similarity from the in-structure distance $d(dc_1, dc_2)$. Note that here we obtain a bound and normalized value, based on an unbound distance measure. The *Type Score Penalty* parameter in (8) is used to express how the absolute distance between two concepts translates into a normalized value.

Algorithm 3: sim_{d-d}, acquired injury-to-acquired injury similarity measure

```
1: assume
   - two acquired injuries $dc_1$ and $dc_2$
2: if $d(dc_1, dc_2) = \infty$ then $sim_{d-d}(dc_1, dc_2) = 0$.
3: else
   $sim_{d-d}(dc_1, dc_2) = \delta^{d(dc_1, dc_2)}$   (8)
   where:
   - $d(dc_1, dc_2)$ is defined in Algorithm 2
```

```
    - $\delta \in \mathbb{R}^1_0$ is a parameter called *Type Score Penalty*
4: end if
```

Algorithm 4 produces a compensation factor c_{s-s} between users in different states. One must expect that users in identical states behave similarly. However, it is also possible to draw some limited conclusions from similar users in different states. We designed the compensation factor with diminishing values according to the distance between the states to highlight this property.

When calculating the compensation factor between states e_1 and e_2 we distinguish the first state (the input state) and the second (the base state). The naming convention was established as we tend to iterate the input state, which is bound to the current user, over all the possible states in order to obtain a list of factors, which later serve as weights in the final similarity calculation. Due to this distinction, the compensation factor is not symmetric:

$$c_{s-s}(e_1, e_2) \neq c_{s-s}(e_2, e_1) \tag{9}$$

The complexity of the Algorithm 4 is a consequence of the most crucial requirement imposed on c_{s-s}. Namely, the sum of all the factors must be constant when iterating one input value over all the possible base values. This condition allows us to use the calculated factors as weights in (12). In addition, the coefficients must decrease in the function of state-to-state distance. Factors linearly dependent on the distance would cause central states to receive higher quality measures than extreme states, simply due to lower average distance. For instance, in the case of three states, the middle one has an average distance of 1 to all the others, while the extreme ones are exactly at a distance of 1.5. Therefore, the use of linear compensation as weights is not recommended, as it favors the central state. The same can be said for all polynomially-dependent coefficients. With our approach we obtain an unbiased coefficient matrix of an exponential nature (see Table 1 for an example) that permits us to take into account diminishing impacts of some states over others. Algorithm 4 takes 2 as the base of the exponential dependency, but it can be easily reformulated for other values.

Algorithm 4: c_{s-s}, state-to-state compensation factor

```
1: assume
    - states $e_i$ (input state) and $e_b$ (base state) with numerical values of $|e_i|$ and $|e_b|$
    - $E$ the number of possible states
2: if $|e_i| = |e_b|$ then $c_{s-s}(e_i, e_b) = 1$
3: else
4:     if $|e_b| > \frac{1}{2}(E-1)$ then $x_1 = E - |e_b|$
5:     else $x_1 = |e_b|$
6:     end if
7:     $x_2 = E - x_1$
```

Table 1: Example of Algorithm 4.

			Base					**Sum**
	State		e_1	e_2	e_3	e_4	e_5	
		Value	0	1	2	3	4	
Input	e_1	0	1.00	0.53	0.27	0.13	0.07	2.00
	e_2	1	0.36	1.00	0.36	0.18	0.09	2.00
	e_3	2	0.17	0.33	1.00	0.33	0.17	2.00
	e_4	3	0.09	0.18	0.36	1.00	0.36	2.00
	e_5	4	0.07	0.13	0.27	0.53	1.00	2.00

```
8:      $C = (2 - 2^{-x_1} - 2^{-x_2})^{-1}$
9:      **obtain**
            $c_{s-s}(e_i, e_b) = C \times 2^{-||e_i|-|e_b||}$        (10)
10: **end if**
```

With the above algorithms in place we may now define a measure for the user-to-test similarity (Algorithm 5) that summarizes all the properties of a test pattern and a user. The value obtained from the algorithm is used as the static component in the posterior quality analysis.

Algorithm 5: sim_{u-t}, user-to-test similarity measure

```
1: assume
    - user $u$ in the state $e_u$ with the acquired injuries: $dc_u = [dc_{u1}, dc_{u2}, \dots]$
    - test $t$, created under the scope of the test pattern $p$, with sets of benefits:
      $dc_p = \{(e_1, \{dc_{11}, dc_{12}, \dots\}), (e_2, \{dc_{21}, dc_{22}, \dots\}), (e_3, \{\dots\}), \dots\}$.
2: for each $(e_k, D_k) \in dc_p$, obtain the partial similarity component $psim_{u-t}$:
    $$psim_{u-t}(dc_u, D_k) = \sum_{dc_i \in dc_u} \sum_{dc_j \in D_k} \gamma^{i-1} \times sim_{d-d}(dc_i, dc_j) \qquad (11)$$
    where:
    - $\gamma \in \mathbb{R}_0^1$ is a parameter named Benefit Score Penalty
    - $i$ iterates over all the acquired injuries of the user $u$ in the order provided
    - $j$ iterates over all the benefits the test pattern $p$ provides for users in the
      state $e_k$
    - $sim_{d-d}$ is the acquired injury-to-acquired injury similarity measure (see
      Algorithm 3)
3: obtain
    $$sim_{u-t}(u, t) = \sum_{(e_k, D_k) \in dc_p} psim_{u-t}(dc_u, D_k) \times c_{s-s}(e_k, e_u) \qquad (12)$$
```

Equation 12 should be understood as a weighted sum of all the partial similarity components $psim_{u-t}$, where the state-to-state compensation factor c_{s-s} serves as a weight. The partial similarity components (11) are the total effect of all the possible cross-combinations of the user's acquired injuries and the test pattern's benefits, with adequate weights in form of γ. We propose the following example for clarification. Assume i) user u in the state e_1 and acquired brain injuries $dc_u = [dc_1, dc_2, dc_3, dc_4, dc_5]$; ii) test pattern p, with benefits $dc_p = \{(e_1, D_1), (e_2, D_2)\} = \{(e_1, \{dc_1, dc_3, dc_4\}), (e_2, \{dc_2, dc_4\})\}$; iii) $sim_{s-s}(e_1, e_2) = 0.85$

Based on these assumptions, in Table 2 we perform a step-by-step calculation of the two possible partial similarity components. With the $psim_{u-t}$ values obtained there, and taking $\gamma = 0.5$, we can calculate $sim_{u-t}(u, t)$ as: $sim_{u-t}(u, t) = (0.6\gamma^4 + 1.1\gamma^3 + 1.1\gamma^2 + 1) \times sim_{s-s}(e_1, e_1) + (0.5\gamma^4 + 1.1\gamma^3 + 0.1\gamma^2 + \gamma) \times sim_{s-s}(e_2, e_1) = 1.45 \times 1 + 0.675 \times 0.85 = 2.05$, which is considered a close similarity.

The final algorithm (Algorithm 6) is the evaluation that ants perform in each node n_i. First note that we divide the components of the evaluation into two groups: the static components $s_{stat}(u, t)$ and the variable components $s_{var}(u, Ts)$. As the name suggests the *static* components hardly ever evolve in time, they are based on the invariants of the users and the test patterns. The age of the user, their profession or other parameters are not absolutely fixed,

Table 2: Example of Algorithm 5.

		dc_u				
		dc_1	dc_2	dc_3	dc_4	dc_5
D_1	dc_1	1	0*	0*	0*	0*
	dc_3	0*	0*	1	0.1*	0.3*
	dc_4	0*	0*	0.1*	1	0.3*
Sum		1	0	1.1	1.1	0.6
Weight		γ^0	γ^1	γ^2	γ^3	γ^4
Weighted Sum		1	0	$1.1\gamma^2$	$1.1\gamma^3$	$0.6\gamma^4$
$psim_{u-t}(dc_u, D_1)$		$0.6\gamma^4 + .1\gamma^3 + 1.1\gamma^2 + 1$				
D_2	dc_2	0*	1	0*	0.1*	0.2*
	dc_4	0*	0*	0.1*	1	0.3*
Sum		0	1	0.1	1.1	0.5
Weight		γ^0	γ^1	γ^2	γ^3	γ^4
Weighted Sum		0	γ	$0.1\gamma^2$	$1.1\gamma^3$	$0.5\gamma^4$
$psim_{u-t}(dc_u, D_2)$		$0.5\gamma^4 + 1.1\gamma^3 + 0.1\gamma^2 + \gamma$				

* example value

and, therefore, even this component may change. The variable component, however, is highly dynamic. It is based on the test solutions Ts present at a given moment in the node n_i.

The equations (14) and, especially, (15) require additional comment. First note that (14) is the static score component multiplied by a geometric average of all the variable components. We opted for the geometric average rather than arithmetic average due to the fact that the data in this case is of multiplicative, not additive nature. In (15) we single out two clauses. The clause (a) increases the impact of users similar to the user u. The second component of this equation is (b) which acts as a score normalizer. It smoothens and softens the linear score dependency and emphasizes higher scores. The $2\pi^{-1}$ factors convert the value range of the subcomponents to $(-1, 1)$ from $(-\pi/2, \pi/2)$. The final $+1$ brings the value to the $(0, 2)$ range. As we are not dealing with probabilities, there is no need to normalize the value to the range $(0, 1)$.

Algorithm 6: Query Evaluation in Node n_i

```
1: assume
     - ant $A$ searches for tests for the user $u$
     - node $n_i$ corresponds to the test $t_i$, within the test pattern $p$
     - set of test solutions $Ts = \{(t_i, u_1, d_1, s_1), (t_i, u_2, d_2, s_2), \ldots, (t_i, u_j, d_j, s_j), ...\}$,
       of size $|Ts|$ which encompasses the performance history of all the past
       solving of test $t_i$.
```

2: **let**

$$s_{stat}(u,t_i) = sim_{u-t}(u,t_i) \tag{13}$$

where $sim_{u-t}(u,t_i)$ is the static score component, defined in Algorithm 5

3: **obtain** score $s_i(u,t_i)$ of the test t_i for the user u:

$$s_i(u,t_i) = s_{stat}(u,t_i) \times \sqrt[|Ts|]{\prod_{Ts_j \in Ts} s_{var}(u,Ts_j)} \tag{14}$$

where $s_{var}(u,Ts_j)$ is the variable score component, calculated for every test solution $Ts_j = (t_i, u_j, d_j, s_j)$ available for the test t_i:

$$s_{var}(u,Ts_j) = \underbrace{\left(2\pi^{-1} sim_{u-u}(u,u_j)\right)^3}_{\text{(a)}} \times \underbrace{\left(2\pi^{-1}\tan(2s_j - 1)\right)}_{\text{(b)}} + 1 \tag{15}$$

4: **let** the evaluation of the node n_i be the tuple $res_i = (n_i, s_i(u,t_i))$

5. Experimental Study

5.1. Experimental Procedure

In order to make mass experiments economically feasible we decided to generate the sets of tests and sets of users in a predetermined and probabilistic manner. The set of users is created according to the strictest rules provided by [35]. Both age-wise, gender-wise and injury-wise distributions are drawn from the mentioned source, which is the most complete we were able to locate. The professions are drawn from the classification [31]. In this way, we argue, one might obtain a statically justifiable set of users. The test patterns are taken directly from our previous work in [29]. Each time we require a set of tests of a given size we generate them maintaining even distribution among the test patterns, i.e. we must expect approximately similar amounts of tests in each test pattern. For instance, in a test graph of 10^4 tests, with 23 test patterns, we have about 430 tests per test pattern, which fulfills the requirements of small-world network. This stochastic approach to the test and test pattern generation will assure that our performance is not the outcome of the test-set employed, which in turn allows drawing general conclusions about the results obtained.

In Table 3 we summarize the ACO-related execution parameters. The parameters are at their standard values, taken from literature [33] and our prior research [36] [37] [32].

5.2. Experiment 1: Zero-knowledge correctness

The first experiment was designed to confirm the correctness of the model and its implementation. In its uninitialized and uninformed state (zero-knowledge state), without pheromone trails or stored test solutions, the system must behave in a fairly straightforward manner, namely, it should be able to find and positively evaluate the theoretically optimal test patterns in a large number of

Table 3: ACS Execution parameters.

ACO		
Parameter	Interpretation	Value
q_0	Weight of exploiting vs. exploring strategy	0.80
α	Pheromone deposition parameter	0.07
ρ	Pheromone evaporation parameter	0.10
β	Weight of link costs	1.00
γ	Weight of evaporation	0.02
τ_{min}	Minimum pheromone level	0.001
τ_{max}	Maximum pheromone level	1.000
τ_0	Initial pheromone level	0.009

queries. Any deviations from this are only allowed once the system has stored enough information to incorporate additional factors.

This experiment was designed in the following way: 1) generate a test world of n tests and 40 users; 2) select a random user along with his 3 theoretically most adequate test patterns; 3) perform 100 queries; in each recommendation list obtained, save the best position of any of the 3 most adequate test pattern. No test solving takes place.

The steps $1 - 3$ were repeated 200 times, 100 for $n = 5 \times 10^3$ and 100 for $n = 10^4$. The experiment-wide average of test recommendations is presented in Fig. 5. The theoretically best test patterns was nearly always placed on top 10 ($\sim 97\%$ of the cases) and, in a majority of cases, it was evaluated as the best one ($83\% - 89\%$). From these results we can conclude that it is highly improbable that a new user will fail to receive optimal suggestions. The suggestions are, naturally, subject to change according to the performance history of the user and the behavior of others. All the aforementioned phenomena are examined in subsequent experiments.

5.3. DNA Graphs

From the user's perspective each query is a simple input-output operation. As input we take the querying characteristics of the user and as output a list of test patterns in descending order of quality. An adequate representation of the system's evolution presented us with a challenge, seeing how the results are multidimensional and change over time. Traditional line graphs were illegible due to the amount of dimensions in question, which in our case surpasses 40. Faced with these challenges we opted for a novel way of representing the evolution that capitalizes on the two dimensions available, in combination with gray-scale intensity. This way we created a visual tool to represent any d-dimensional value evolving over n_{max} discrete time units.

We refer to this type of graphs as *dna-graphs* (see Fig. 6)), as they resemble quite closely dna test results. Reading the dna-graphs is straightforward. First, the time axis runs, traditionally, left to right; $n = 0$ is the leftmost edge, while the rightmost one is the end of the experiment. Each vertical cut is the set of

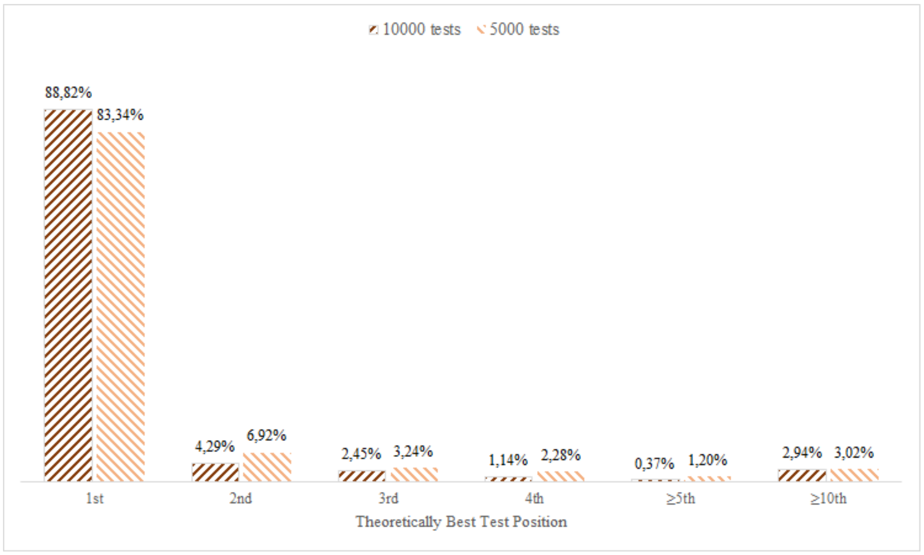


Figure 5: Zero knowledge state correctness.

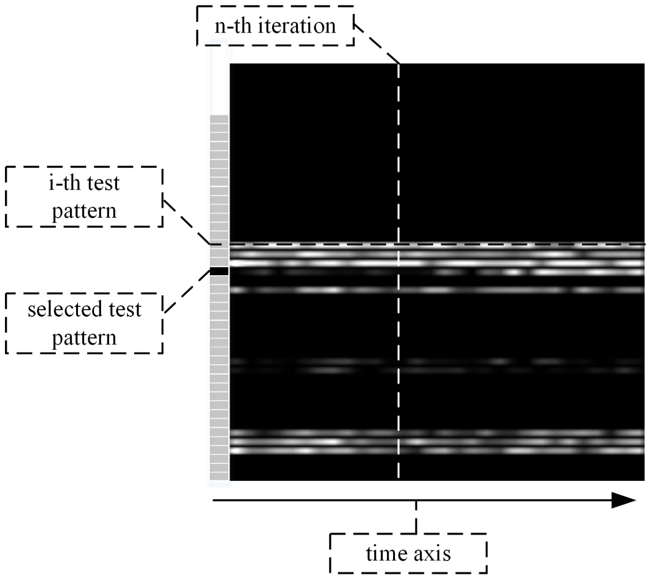
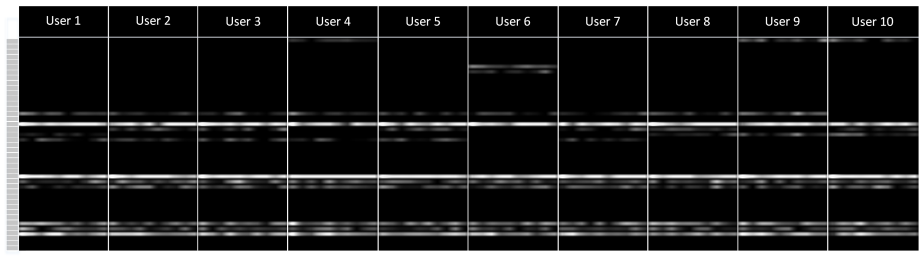
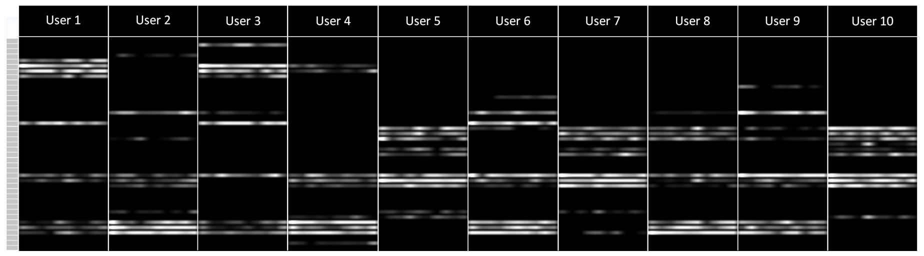


Figure 6: DNA-Graph example. The parallel evolution of the rank (gray-scale intensity) of 46 test patterns (vertical axis) in function of time (horizontal axis).

recommendations in the n-th time unit, also referred to as the n-th iteration. Each horizontal cut represents the evolution of a single test pattern. Finally, the gray-scale intensity represents the position of the test pattern in the recommendation list - the brighter it is, the higher on the list. Black areas represent



(a) Similar Users



(b) Random Users

Figure 7: User Groups. System outputs for different groups of users.

test patterns absent from the list. In order to facilitate the graph reading we transferred the idea of rolling average into the visual representation as motion blur, which runs left to right, along the time axis. Note the black rectangle on the left side. On some occasions we use it to mark a test pattern of special significance, the rest will be left in gray.

5.4. Experiment 2: Inter-user Similarity

In this experiment we analyze the correctness of the user-to-user (sim_{u-u}) similarity measure. Two similar users must receive approximately the same response from an uninitialized system, i.e. before any tests have been solved or pheromone spread and additional profiling could take place.

We generate the test world with 10^3 tests and 10 users, with the prerequisite that the similarity between each pair of users must not be inferior to 0.85, and we have each user performs 100 queries. There is no test solving involved, similarly to Experiment 1, we only observe the responses of the system.

As can be clearly seen in the series of 10 dna-grahps in Fig. 7a, users have received a very similar set of test patterns, with only minor deviations. For instance, User 6 receives some additional test patterns, whereas the rest do not. These differences confirm that the system is capable of distinguishing between even very similar users, yet maintaining the cohesion of results to a high degree. On the contrary, Fig. 7b is an example of a group of random users with their different test pattern recommendation.

Statistical Analysis

Goal. Demonstrate that the User-to-User similarity corresponds to an actual similarity of the recommendations. We chose the Pearson Correlation, which is commonly used in such cases.

Procedure. First we generate a pool of 100 users with random pairwise similarities, next we let each user perform 100 queries in order to obtain for each one a stable dna-graph of responses, as in Fig. 7. Then we process our data crosswise obtaining 10^5 pairs of (User-to-User similarity, Result-to-Result similarity), where the Result-to-Result similarity is the pixel-by-pixel overlap measures of two dna-graphs. Note that it is an inverse scale, which means that the perfect similarity between graphs is evaluated as 0.

Results. Our results show that there is a strong Pearson Correlation ($Pc = -0.423$, $Sig < 0.01$) between the two types of similarity, which in turn tells us that the user-to-user similarity measure is correct; i.e. similar users will receive similar recommendations, while different users will receive increasingly different recommendations.

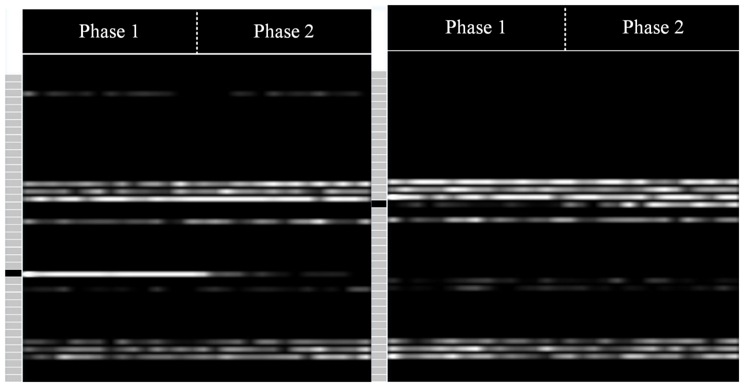
5.5. Experiment 3: Unexpected Good and Unexpected Bad

In Experiments 1 and 2 we have shown that users are initially given a reasonable set of tests and similar users receive similar recommendations. In this experiment we chose to see how the system reacts to unorthodox behavior, i.e. one that contradicts the base profile of a user. The design of the experiment was the following:

1. Take a single user.
2. Perform 50 queries, no test solving (Fig. 8, Phase 1).
3. (Unexpected Bad variant) Select one of the top evaluated test patterns and solve it very badly 50 times. Score 10% of the maximum score. (Fig. 8a, Phase 2).
4. (Unexpected Good variant) Select one of the bottom evaluated test patterns and solve it very well 50 times. Score 95% of the maximum score. (Fig. 8b, Phase 2).

What we observe is that the system quickly corrects the recommendation lists, incorporating the newly gained, user-related knowledge.

In the Fig. 8a we can see that, starting from the halfway point, the test pattern in question immediately drops from the top position to one of the last, eventually disappearing from the results completely. The opposite behavior is witnessed in Fig. 8b where one of the worst of the top ten test patterns is suddenly elevated, after being solved well several times. Note the black rectangle on the left side of both subfigures in Fig. 8 that indicates the test pattern in question.



(a) Unexpected bad **(b)** Unexpected Good

Figure 8: Unexpected Behavior. Reinforcement or removal of test patterns.

Statistical Analysis

Goal. Demonstrate that the unexpected behavior results in significant changes in the recommendation lists. We chose the Mann Whitney U test, as it is the most commonly used in non-parametric tests with two populations.

Procedure. We decided to construct the test focusing on the recommended test pattern relative rank shift. This rank shift was calculated as a difference between the mean rank of the test pattern during the first 50 iterations and the last 50 iterations, i.e. if a test pattern was 2nd on the recommendation list and it changed with time to the 6th position, then we would consider $RankShift = -4.0$; if it changed from the 5th position to the 3rd position, then $RankShift = 2.0$, and so on. In both variants, we tracked the evolution of 10 test pattern recommendations, 90% of which were left unsolved (Mode = Unaffected) and 10% solved in an unexpected manner (Mode = Unexpected), all repeated 100 times.

Hypothesis. Mann-Whitney U test.

Dependent Variable. *RankShift* is defined as $RankShift = \overline{Rank_{51-100}} - \overline{Rank_{1-50}}$, where $\overline{Rank_{j-i}}$ is the average rank of the given test pattern between iterations i and j.

Independent Variables. *Mode* is defined as *Unaffected* for unaffected test patterns and *Unexpected* for tests solved in either unexpectedly well or unexpectedly badly.

Null Hypotheses. H_0 the Mode variable does not influence the $RankShift$, H_1 the Mode variable does influence the $RankShift$.

Results. In the Unexpected Good variant we observed the mean $RankShift$ of 0.27 for unaffected tests and -4.1 for tests solved well, with test statistics $U = 5.5$, $Sig < 0.001$. Similarly, in the Unexpected Bad variant we

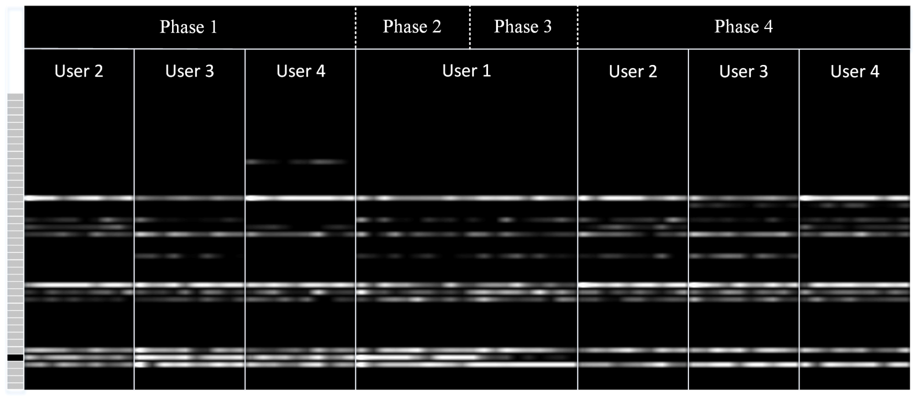


Figure 9: Inter-user influence. Indirect influence of User 1 on Users 2 - 4.

observed the $RankShift$ of -0.22 and 4.25 for unaffected tests and test solved badly, respectively, with test statistics $U = 1.0$, $Sig < 0.001$. In both variants we must reject the null hypothesis and assume that the fact of solving tests in an unexpected manner affects the test ordering in a significant way and, therefore, that the system is capable of adjusting itself to the user's progress.

5.6. Experiment 4: Indirect Inter-user Influence

In this experiment we intended to show how the behavior of one user can affect a whole cluster of similar users in an indirect manner. A test world with 10^3 tests and 4 users, with high degree of pairwise similarity (> 0.85) was generated. We choose a central user, with the highest average similarity to other users and label him user 1; others become user 2, user 3 and user 4. The experiment consists of the following steps:

1. Users 2 - 4 perform 100 queries, without solving tests (Fig. 9, Phase 1).
2. User 1 performs 100 queries without solving tests (Fig. 9, Phase 2).
3. The best common test pattern for all the Users is chosen.
4. User 1 performs 100 queries, each time solving badly the previously chosen pattern (Fig. 9, Phase 3).
5. Users 2 - 4 repeat the 100 queries, without solving tests (Fig. 9, Phase 4).

Note that during the run of this experiment only user 1 solves tests.

Fig. 9 shows the responses of the system. We can see how the test pattern marked with a black rectangle on the left side was completely removed from the suggested list of all the users after only one member of the user group solved it badly.

Statistical Analysis

Goal. Demonstrate that unexpected behavior of a user significantly affects the recommendation lists of closely similar users. We chose the Mann Whitney U test, as in Experiment 3.

Procedure. The statistical analysis in this experiment was similar to the one in Experiment 3. We generated 20 groups of 4 users and performed the full experiment 20 times. To every test pattern for every user we assigned the $RankShift$ value, which is the difference between the average rank of the test pattern in Phase 1 and Phase 4 and the Mode value. The Mode takes two values: *Affected* in case of the test pattern in question and *Unaffected* in every other case.

Hypothesis. Mann-Whitney U test.

Dependent Variable. *RankShift* is defined as $RankShift = \overline{Rank_{phase4}} - \overline{Rank_{phase1}}$, where $\overline{Rank_{phaseN}}$ is the average rank of the given test pattern in the phase n.

Independent Variables. *Mode* is defined as *Unaffected* for unaffected test patterns and *Affected* for tests solved by the User 1.

Null Hypotheses. H_0 the Mode variable does not influence the $RankShift$, H_1 the Mode variable does influence the $RankShift$.

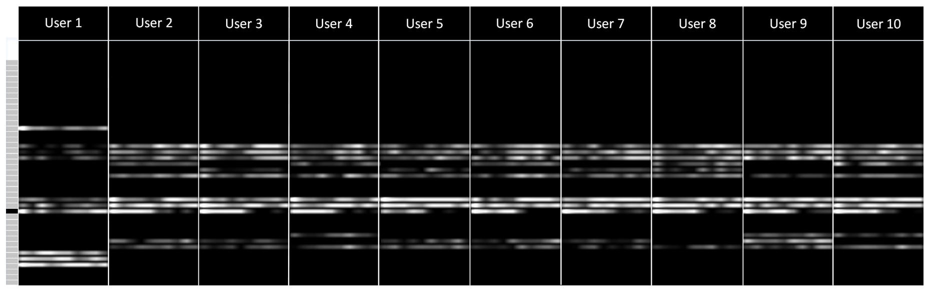
Results. Using the Mann-Whitney U test we concluded that there was a statistically significant difference in rank changes between the Mode of the two groups, $U = 8.0$, $Sig < 0.01$. The average RankShift was -0.08 and 3.91 for the Unaffected and Affected group, respectively. We therefore must reject the null hypothesis and assume that the recommendation for test patterns solved by the central user was affected significantly for all the users involved in the experiment.

It is now obvious that similar users affect each other. However, this could cause unwanted results, if the behavior of the user *constantly* becomes dominated by the behavior of others. In order to examine this question we proposed Experiment 5.

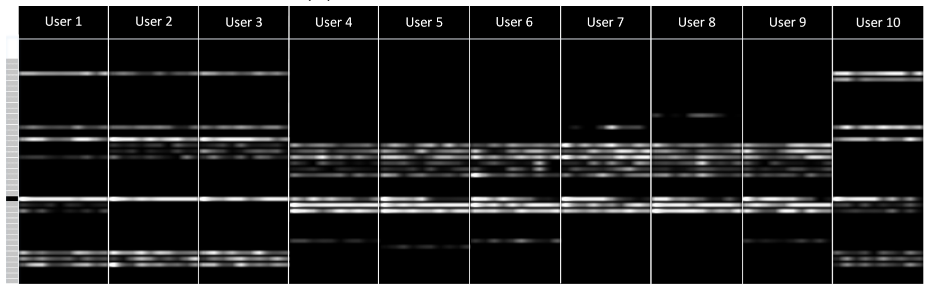
5.7. Experiment 5: Fine-scale user clustering

The previous experiment answers the question about what happens if a subset of a group of similar users acts in an unexpected manner. Here we try to show a more difficult case when similar users act in contradictory ways. In other words, we explore to what degree the system can distinguish between users if their calculated similarity measure does not coincide with their actual behavior.

We assume that 10 similar users exist (pairwise similarity > 0.80), but behave in two distinct ways. First, there is a group of *Good Users* that solve the queried tests as expected, i.e. theoretically suitable tests are solved well and the unsuitable ones badly; second, there is a group of *Bad Users* that do the exact opposite. These are the phases in the experiment:



(a) 1 Good User, 9 Bad Users



(b) 3 Good Users, 7 Bad Users

Figure 10: Fine-scale clustering. Subclustering of similar users under the condition of contradictory behavior

1. All users query test patterns, without solving, for the first half of the experiment.
2. The best common test pattern is chosen.
3. Good Users start solving well the test within the chosen test pattern, while Bad Users start solving them badly.

Fig. 10 is a visualization of the responses for: A) 1 Good User and 9 Bad Users; B) 3 Good Users and 7 Bad Users. As we can see, in spite of the high level of similarity, the system was capable of modifying its behavior in two opposite ways. All Good Users still receive the test pattern in question, while all Bad Users have it pushed off the first place on the list to a point at which it almost disappears. In both figures the test patterns affected are marked with a black rectangle on the left. This fine level split proves that the system can detect subgroups of users based on their behavior as well as static characteristics.

5.8. Experiment 6: Global Experiment

After analyzing the system in a series of isolated situations, in which the outputs were foreseeable and the inputs had direct consequences, we proceed to a global experiment. In this final experiment we would like to examine the performance of the system under a more realistic situation of a mixture of the aforementioned behaviors. We also allow users to evolve, change their behavior modes and attitudes, and we incorporate several physiological components such as boredom and curiosity.

There is an inherent difficulty in performing a global, realistically simulated experiment. The more effort that is put into modeling natural user behavior, the more difficult it is to predict the outputs of the system, which, in turn, makes the statistical analysis less straightforward.

In order to accurately model the users, we need to introduce a new concept, *User Mentality*, which in our model has the following values: Passive, Neutral, UnexpectedGood, UnexpectedBad and Random. They can be understood as follows:

Passive. Users with this mentality are only querying for tests, but never solve them. It should be perceived as initial curiosity in the system, combined with a certain degree of timidity. Every user starts with Passive mentality.

Neutral. These users solve all the tests according to the quality provided by the model. This means that they should have a neutral impact on the evolution of the system. Having said that, as we show in the Experiment 4, they can be influenced by other users. When a User chooses to evolve from the Passive mentality, it always changes first to Neutral.

UnexpectedGood and UnexpectedBad. These two mentalities are identical to those in Experiment 3. With a certain probability users with Neutral mentality can choose one Test Pattern as the objective of their unexpected behavior. A test from the recommendation rank 5 - 10 can be selected as the objective of UnexpectedGood mentality, and one from rank 1 - 3 can become the objective of an UnexpectedBad mentality. Each time a test is solved in an unexpected manner we increase or decrease the accumulated *Impact* the users had on it ($+1$ for UnexpectedGood solution, -1 for UnexpectedBad solution).

Random. The last mentality is random behavior. Neutral, UnexpectedGood and UnexpectedBad users can start and behave irrationally, solving tests in a random manner. This sort of mentality corresponds to bored or annoyed users, as well as input errors. It serves as a way of establishing how well the system copes with input noise. Random users can return to behaving in a Neutral way.

The transitions between states are probabilistic and expressed in the evolution matrix E_M, (16). The probability of the transition from mentality m_1 to m_2 is in the the m_1-st row and m_2-nd column, $E_M[m_1][m_2]$. For instance: the probability for Neutral (2) to convert to UGood (3) is $E_M[2][3] = 0.01$.

$$E_M = \begin{array}{c} \\ Passive \\ Neutral \\ UGood \\ UBad \\ Random \end{array} \begin{array}{c} \begin{array}{ccccc} Passive & Neutral & UGood & UBad & Random \end{array} \\ \begin{pmatrix} - & 0.03 & 0 & 0 & 0.002 \\ 0 & - & 0.01 & 0.01 & 0.002 \\ 0 & 0 & - & 0 & 0.0001 \\ 0 & 0 & 0 & - & 0.0001 \\ 0 & 0.008 & 0 & 0 & - \end{pmatrix} \end{array} \tag{16}$$

Statistical Analysis

As the results of this experiment are too complex to visualize by the dna-graphs, we therefore only perform a statistical analysis.

Goal. Demonstrate that in a non-isolated, global experiment the recommendation list ranks are correlated with the behavior of users. We chose the Pearson Correlation Pc, as in Experiment 3.

Procedure. For each experiment run we generate a pool of 10^3 tests under 46 tests patterns and U users grouped in C clusters. Users from within a cluster have pairwise similarity > 0.8, users from different clusters have pairwise similarity of < 0.6. In addition, we take the evolution matrix as E_M/R, where R is referred to as *Randomness decrease*, i.e. the higher the R, the less probable the evolution, the less users behave in an unexpected manner and the lower the Impact value for each test pattern.

We have split this experiment into two variants. Variant i) *uniform*, in which the evolution of the entire population is uniform and determined by R; ii) *per-cluster* in which the evolution of each cluster of the population is determined by a different R_C. One experiment run consists of 10^3 iterations. A full iteration is composed of: i) selecting a random user from U; ii) performing a query for tests for the selected user; iii) handling the query results according to the selected user's mentality; iv) evolving all users in U with probabilities given by E_M/R (*uniform* variant) or E_M/R_C (*per-cluster* variant) v) saving the ranks of all the test patterns queries and updating the impacts, if the solution was performed in an unexpected manner.

Each experiment run is repeated five times for the same U, C and R. In our experiment we took $U = \{10, 25, 50, 100\}$, $C = \{1, 2, 3, 5\}$, $R = \{1, 10, 50, 100\}$ and $R_C \in \{1, 10, 50, 100\}$, which resulted in 64 combinations, $5\times64 = 320$ experimental runs for each variant and 640 experimental runs total. The statistical analysis is reported for all the aggregated results Pc, as well as, grouped for each R: $Pc[R = 1]$, $Pc[R = 10]$, $Pc[R = 50]$, $Pc[R = 100]$ for both variants.

Hypothesis. Pearson Correlation Pc test.

Dependent Variable. *RankShift* is defined as $RankShift = \overline{Rank_{0-50}} - \overline{Rank_{950-1000}}$. It is the difference in average rank of a given test pattern in the first 5% and the last 5% iterations.

Table 4: Experiment 6. Pearson Correlation results.

	Uniform		Per-cluster	
Parameter	Pc	Sig	Pc	Sig
Pc	-0.469^{**}	< 0.00	-0.352^{**}	< 0.00
$Pc[R = 1]$	-0.652^{**}	< 0.00	-0.503^{**}	< 0.00
$Pc[R = 10]$	-0.451^{**}	< 0.00	-0.290^{**}	< 0.00
$Pc[R = 50]$	-0.146^{**}	< 0.00	-0.168^{**}	< 0.00
$Pc[R = 100]$	-0.094^{**}	< 0.00	-0.143^{**}	< 0.00

** Correlation is significant at the 0.01 level (2-tailed).
* Correlation is significant at the 0.05 level (2-tailed).

Independent Variables. *Impact* is calculated as an absolute sum of all the unexpected solutions, tracked independently for each test pattern.

Null Hypotheses. H_0 the Impact variable does not influence the RankShift, H_1 the Impact variable does influence the RankShift.

Results. In the *uniform* variant the aggregated results, as well as each of the results grouped by Randomness decrease, have been shown significant, see Table 4. We must therefore conclude that Impact variable influences RankShift in an expected manner in the *uniform* execution variant, confirming Experiments 3 and 4 in the global setup. In addition, we see that the Impact is inversely proportional to RankShift and decreasing with R.

If the evolution of the system is not uniform (*per-cluster*) we also obtain statistically significant results, see Table 4. This means that even if the evolution, and consequently the behavior of the users of the system, varies from cluster to cluster we still obtain statistically significant recommendation list changes, confirming Experiments 3 and 5 in the global setup. The Impact variable is again inversely proportional to RankShift and decreasing with R

We therefore conclude that our recommendation system works as expected under all conditions.

Aside from the main statistical study we performed an analysis of the distribution of the Impact variable, grouped by R (Fig. 11). We chose R as the grouping variable, as it is the main factor in the generation of User Mentalities. We can clearly see the mean in all the cases remains very close to 0, which is desirable and expected. The user pool must behave unexpectedly in a small portion of the iterations, rather than a majority. The User population generated with $R = 1$ (Fig. 11a) is the most quickly evolving and, in consequence, generates the most impacts. The standard deviation of the Impact variable distribution is 33.126. As we proceed towards slowly evolving populations, the standard deviation decreases, reaching 2.014 in the Fig. 11d. This suggests that our model of user and user mentality generation is reasonable and realistic.

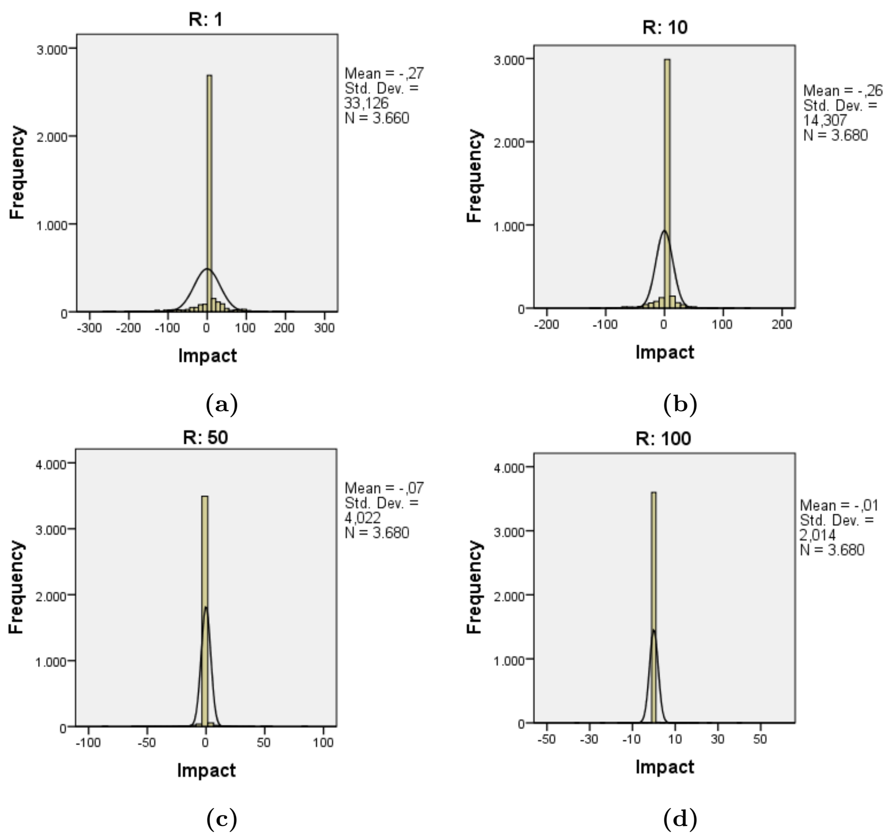


Figure 11: The distribution of the Impact. Grouped by R

6. Conclusions and Future Work

In this work we have shown that our proposal proved to be appropriate for the problem in question. ACO-based suggestion building is efficient starting from the zero-knowledge stage, achieving precision of 80% - 90%. This indicates that any user approaching the system would immediately receive quality suggestions, as demonstrated in Experiment 5.2. In later experiments we have shown conclusively that the precision only improves from that point.

The inter-user indirect influence is akin to ant communication in ACO models and is, therefore, deemed the most suitable metaphor. An important property of ACO is the ability to reconverge if the desired conditions arise. Naturally, the users' behavior can evolve gradually, but it can also change abruptly at any point in time. Such an event causes a quick response in the form of a reevaluation and reconvergence with system-wide consequences. Experiment 3 demonstrates precisely this situation. In addition, Experiment 4 shows the wider consequences of such an event. Here a group of users starts receiving significantly different suggestions based purely on the actions of others.

Apart from its adaptability the suggested model has the potential to distinguish fine differences between apparently similar users. This means that any groups that may have formed over the evolution of the system can be very quickly split into sub-groups if the conditions so dictate. Our proposed distinction between the static and dynamic score components facilitates this process. This fine scale similarity reevaluation was shown in Experiment 5.

Finally, we performed a global experiment (Experiment 6), the largest in terms of iterations and user-pools, which was designed to integrate all the partial views provided by Experiments 1 - 5. In this experiment we maintain a dynamic, evolving user population with different behaviors, ranging from Passive to Random, with the full range of disabilities, and in all possible physical states. We consider that in this way we achieve an acceptably accurate representation of a set of real-life users. From Experiment 6 we conclude that our model reacts as expected to user actions and generates statistically relevant recommendation lists.

Several challenges constitute our future work. Currently, we are working on the integration of the solution here presented into HABITAT, the application developed for the treatment of individuals with ABI in order to enable them to take full advantage of our findings. This would involve a step forward in their treatment and reduce the need for specialist attention. This integration must be carefully planned, not because of technological issues, but to control its impact on the evolution of system users. Therefore, a careful deployment plan must be designed to maximize the acceptability of the proposal in clinical terms.

It would be of great interest to analyze if other well-known metaheuristics are equally applicable in this domain as the underlying algorithmic structure. Most notably, it would be interesting to examine the potential of the Particle Swarm Optimization (PSO), which has been shown to have a multitude of applications [38].

In addition, from a purely technological point of view, several issues must be resolved to enable interaction with these individuals. For instance, it would be necessary to design proper facilities to guide them in the query processes, so that they know the real meaning of the minimum and maximum scores. It would also be interesting to model additional factors, such as daily progress and fatigue levels, so that the system can encourage the users to either continue with additional tests or to rest.

Finally, in relation to the algorithm presented here, we are evaluating new meta-heuristics that take into account the stress of the person with ABI while they are using the system. These heuristics are oriented towards providing full support for the personal part of the relearning process. They would also consider additional inputs, not only about the users' cognitive state, but also about their physical conditions while solving the tests. The technique presented here has the potential to be used outside the ABI field. An example reformulation of the presented schema might be used for recommendation of rehabilitation game-based therapies for hospitalized children in pediatric services.

Acknowledgments

This research has been funded by the Spanish Ministry of Economy and Competitiveness and by the FEDER funds of the EU under the project SUPEREMOS (TIN2014-60077-R) and insPIre (TIN2012-34003). Kamil Krynicki is supported by the FPI fellowship from Universitat Politècnica de València.

References

[1] WHO, International classification of impairments, activities and participation (ICIDH2): document for field trial purposes, Tech. rep., World Health Organization, Geneva, Switzerland (1997).

[2] M. Faul, L. Xu, M. M. Wald, V. G. Coronado, Traumatic Brain Injury in the United States: Emergency Department Visits, Hospitalizations and Deaths 2002–2006, Tech. rep., U.S. Department of Health and Human Services (2010).

[3] O. Zaiane, Building a recommender agent for e-learning systems, International Conference on Computers in Education, 2002. Proceedings.doi:10.1109/CIE.2002.1185862.

[4] K. Verbert, N. Manouselis, X. Ochoa, M. Wolpers, H. Drachsler, I. Bosnic, E. Duval, Context-aware recommender systems for learning: A survey and future challenges (2012). doi:10.1109/TLT.2012.11.

[5] M. Nilashi, O. B. Ibrahim, N. Ithnin, Hybrid recommendation approaches for multi-criteria collaborative filtering, Expert Systems with Applications 41 (2014) 3879–3900. doi:10.1016/j.eswa.2013.12.023.

[6] Y. J. Yang, C. Wu, An attribute-based ant colony system for adaptive learning object recommendation, Expert Systems with Applications 36 (2009) 3034–3047. doi:10.1016/j.eswa.2008.01.066.

[7] S. B. Aher, L. M. R. J. Lobo, Combination of machine learning algorithms for recommendation of courses in E-Learning System based on historical data, Knowledge-Based Systems 51 (2013) 1–14. doi:10.1016/j.knosys.2013.04.015.

[8] S. Bakshi, A. K. Jagadev, S. Dehuri, G.-N. Wang, Enhancing scalability and accuracy of recommendation systems using unsupervised learning and particle swarm optimization, Applied Soft Computing 15 (2014) 21–29. doi:10.1016/j.asoc.2013.10.018.

[9] S. Wang, Y. Zhang, G. Ji, J. Yang, J. Wu, L. Wei, Fruit classification by wavelet-entropy and feedforward neural network trained by fitness-scaled chaotic abc and biogeography-based optimization, Entropy 17 (8) (2015) 5711. doi:10.3390/e17085711.

[10] S. G. Li, L. Shi, The recommender system for virtual items in MMORPGs based on a novel collaborative filtering approach, International Journal of Systems Science (2013) 1–16doi:10.1080/00207721.2012.762560.

[11] C. M. Chen, H. M. Lee, Y. H. Chen, Personalized e-learning system using Item Response Theory, Computers and Education 44 (2005) 237–255. doi:10.1016/j.compedu.2004.01.006.

[12] C. De Maio, G. Fenza, M. Gaeta, V. Loia, F. Orciuoli, S. Senatore, RSS-based e-learning recommendations exploiting fuzzy FCA for Knowledge Modeling, Applied Soft Computing Journal 12 (2012) 113–124. doi:10.1016/j.asoc.2011.09.004.

[13] A. Baylari, G. A. Montazer, Design a personalized e-learning system based on item response theory and artificial neural network approach, Expert Systems with Applications 36 (2009) 8013–8021. doi:10.1016/j.eswa.2008.10.080.

[14] P. J. Eslinger, Neuropsychological Interventions: Clinical Research and Practice, The Guilford Press, New York, New York, USA, 2005.

[15] L. De-Marcos, C. Pages, J. J. Martínez, J. A. Gutiérrez, Competency-based learning object sequencing using particle swarms, in: Proceedings - International Conference on Tools with Artificial Intelligence, ICTAI, Vol. 2, 2007, pp. 111–116. doi:10.1109/ICTAI.2007.14.

[16] R. Sharma, H. Banati, P. Bedi, Adaptive content sequencing for e-learning courses using ant colony optimization, in: Advances in Intelligent and Soft Computing, Vol. 131 AISC, 2012, pp. 579–590. doi:10.1007/978-81-322-0491-6_53.

[17] F. H. Wang, On extracting recommendation knowledge for personalized web-based learning based on ant colony optimization with segmented-goal and meta-control strategies, Expert Systems with Applications 39 (2012) 6446–6453. doi:10.1016/j.eswa.2011.12.063.

[18] F.-H. Wang, Personalized recommendation for web-based learning based on ant colony optimization with segmented-goal and meta-control strategies, 2011 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE 2011) (2011) 2054–2059doi:10.1109/FUZZY.2011.6007628.

[19] Y. Zhang, S. Wang, G. Ji, A rule-based model for bankruptcy prediction based on an improved genetic ant colony algorithm, Mathematical Problems in Engineering 2013. doi:10.1155/2013/753251.

[20] Brain Injury Statistics, last visited: 2015-06-03.
URL `http://ca.bicbraininjurycenter.com/`

[21] Key facts and statistics, last visited: 2015-04-17.
URL `https://www.headway.org.uk/key-facts-and-statistics.aspx`

[22] M. Chamberlain, V. Neumann, A. Tennant, Traumatic brain injury rehabilitation: Services, Treatements and Outcomes, in: V. Neumann (Ed.), Principles and practice of treatment, Chapman & Hall Medical, London, 1995, pp. 101–118.

[23] K. D. Cicerone, C. Dahlberg, J. F. Malec, D. M. Langenbahn, T. Felicetti, S. Kneipp, W. Ellmo, K. Kalmar, J. T. Giacino, J. P. Harley, L. Laatsch, P. A. Morse, J. Catanese, Evidence-based cognitive rehabilitation: Updated review of the literature from 1998 through 2002 (2005). doi:10.1016/j.apmr.2005.03.024.

[24] M. S. McKay, C. A. Mateer, Introduction to cognitive rehabilitation: Theory and practice, The Guilford Press, 1989.

[25] K. D. Cicerone, C. Dahlberg, K. Kalmar, D. M. Langenbahn, J. F. Malec, T. F. Bergquist, T. Felicetti, J. T. Giacino, J. P. Harley, D. E. Harrington, J. Herzog, S. Kneipp, L. Laatsch, P. A. Morse, Evidence-based cognitive rehabilitation: Recommendations for clinical practice (2000). doi:10.1053/apmr.2000.19240.

[26] C. Christiansen, B. Abreu, K. Ottenbacher, K. Huffman, B. Masel, R. Culpepper, Task performance in virtual environments used for cognitive rehabilitation after traumatic brain injury., Archives of physical medicine and rehabilitation 79 (1998) 888–892. doi:Doi 10.1016/S0003-9993(98)90083-1.

[27] E. Navarro, V. López-Jaquero, F. Montero, HABITAT: a web supported treatment for Acquired Brain Injured, in: Proceedings - The 8th IEEE International Conference on Advanced Learning Technologies, ICALT 2008, 2008, pp. 464–466. doi:10.1109/ICALT.2008.151.

[28] F. J. Navarro, E. Navarro, F. Montero, HABITAT: A tool for interactive activities in the treatment of Acquired Brain Injury, in: 13th International Conference on Interacción Persona-Ordenador (INTERACCION '12), 2012, pp. 1–2. doi:10.1145/2379636.2379641.

[29] F. Montero, V. López-Jaquero, E. Navarro, E. Sánchez, Computer-aided relearning activity patterns for people with acquired brain injury, Computers and Education 57 (2011) 1149–1159. doi:10.1016/j.compedu.2010.12.008.

[30] M. Dorigo, G. D. Caro, Ant colony optimization: a new meta-heuristic, Proceedings of the 1999 Congress on Evolutionary Computation-CEC99 (Cat. No. 99TH8406) 2. doi:10.1109/CEC.1999.782657.

[31] Clasificación Nacional de Ocupaciones (in Spanish), last visited: 2015-06-03.
URL `http://www.ine.es/clasifi/cno94.xls`

[32] K. Krynicki, J. Jaen, J. A. Mocholi, On the performance of ACO-based methods in p2p resource discovery, Applied Soft Computing Journal 13 (2013) 4813–4831. doi:10.1016/j.asoc.2013.07.022.

[33] M. Dorigo, L. M. Gambardella, Ant colony system: a cooperative learning approach to the traveling salesman problem, IEEE Transactions on Evolutionary Computation 1 (1997) 53–66. doi:10.1109/4235.585892.

[34] E. Michlmayr, A. Pany, G. Kappel, Using taxonomies for content-based routing with ants, Computer Networks 51 (2007) 4514–4528. doi:10.1016/j.comnet.2007.06.015.

[35] L. O'Rance, N. Fortune, Disability in Australia: Acquired Brain Injury, Australian Institute of Health and Welfare, 2007.

[36] K. Krynicki, M. E. Houle, J. Jaen, Angry Ants: A non-Hybrid Ant Colony Optimization Heuristic for Convergence Quality, in: IEEE International Conference On Systems, Man, and Cybernetics, 2015, accepted. Under Publication.

[37] K. Krynicki, J. Jaen, A. Catala, A diffusion-based ACO resource discovery framework for dynamic p2p networks, in: 2013 IEEE Congress on Evolutionary Computation, IEEE, 2013, pp. 860–867. doi:10.1109/CEC.2013.6557658.

[38] Y. Zhang, S. Wang, G. Ji, A comprehensive survey on particle swarm optimization algorithm and its applications, Mathematical Problems in Engineering 2015. doi:10.1155/2015/931256.