Using Context History to Personalize a Resource Recommender via a Genetic Algorithm

Alessandro Ciaramella
IMT Lucca Institute for Advanced Studies,
Piazza San Ponziano 6, 55100 Lucca, Italy
a.ciaramella@imtlucca.it

Mario G.C.A. Cimino, Beatrice Lazzerini,
Francesco Marcelloni
Dipartimento di Ingegneria dell'Informazione:
Elettronica, Informatica, Telecomunicazioni,
University of Pisa,
Largo Lucio Lazzarino 1, 56122 Pisa, Italy
{m.cimino, b.lazzerini, f.marcelloni}@iet.unipi.it

Abstract — **Situation awareness is a promising approach to recommend to a mobile user the most suitable resources for a specific situation. However, determining the correct user situation is not a simple task since users have different habits that may affect the way in which the situations arise. Thus, an appropriate tuning aimed at adapting the situation recognizer to the specific user is desirable to make a resource recommender more effective. In this paper, we show how this objective can be achieved by collecting data during the interaction of the user with the mobile device and using this context history to personalize the resource recommender by a genetic algorithm. To describe our approach, we adopt a recently proposed resource recommender which exploits fuzzy linguistic variables to manage the inherent vagueness of some contextual parameters. Experimental results on a real business case show that the responsiveness and modeling capabilities of the recommender increase, thus validating the proposed approach.**

Keywords - context awareness; context history; genetic fuzzy systems; mobile resource recommender; service personalization.

I. Introduction

Users in mobility can today access to a very large amount of different resources. Thus, to search for the most suitable resource for each specific situation is not typically a trivial task and might be very time-consuming. In our previous works [5][6], we have presented a resource recommender that is able to suggest a list of resources suitable for the specific user task and situation. The situation is automatically recognized from contextual data by appropriately combining a semantic layer with a fuzzy layer. More specifically, the semantic layer is devoted to express domain knowledge through web ontologies and rules, whereas the fuzzy layer is in charge of managing the natural vagueness and impreciseness of contextual data. The combination of the two layers provides a list of situations sorted by decreasing degrees of recognition certainty. Given a specific task, an appropriate ontology links resources to each specific situation [12]. Thus, using the degree of recognition certainty associated with each situation, we can propose to the user a sorted list of resources which are likely to be needed by the user in the specific operation context: the resources at the top and the bottom of the list correspond to the situations recognized with the highest and lowest degrees of certainty, respectively.

In our previous works, we proposed a general architecture for the recommender system and applied this architecture to two case studies, namely a pharmaceutical consultant and an off-site student. We defined the linguistic variables used in the fuzzy layer considering a generic user without taking specific user habits into account. Of course, the definition of these variables through the corresponding membership functions is a critical step of the overall recommending process [14]. Indeed, the shape and position of these functions strongly affect the computation of the degrees of certainty. Thus, to increase the recognition rate, shapes and positions should be adapted to the user habits. Currently, some systems already allow a personalization degree, but the users have to input and update their preferences manually in order to receive personalized services. A more efficient technique for personalization would be to deduce user habits automatically from the context history. Indeed, the employment of context history can be extremely effective in enabling personalization and adaptation by discovering recurrent patterns in the data [3].

In this paper, we briefly describe the architecture of our resource recommender. Then, we show how the definition of the linguistic variables can be tuned to the specific user via a genetic algorithm (GA) by using the context history. Finally, we discuss using a real business case how this personalization increases the performance of our system, allowing recognizing each situation with a higher precision than the system developed for a generic user.

The paper unfolds as follows. Section 2 summarizes the state of the art on the use of context history in recommendation systems. In Section 3, the overall system architecture of the situation-aware resource recommender is presented and discussed. Section 4 describes in detail the implementation of the proposed approach, whereas Section 5 presents the application of the proposed approach to a real case study. Finally, in Section 6, some concluding remarks are reported.

II. Related Work

In the field of recommendation systems, context can be defined as any information that characterizes the user situation in which a need of resources arises [11]. Hence, determining the user situation is crucial to recommend the most suitable resources. Context history has been identified as an important piece of information to recognize the user situation. Context history is strictly related to the activity that the users are going to perform, and to the resources which they might be interested in [3]. However, the use of context history in recommendation systems is considered a relatively under-explored area [9]. Maryhofer has proposed in [13] to use context history to predict the current situation. Here, sensor data are classified into higher-level context identifiers, and then the next possible contexts are predicted by using an algorithm based on Markov models. Byun and Cheverst have exploited context history to induce rules for adapting the system to the user behavior [3][4]. In particular, fuzzy decision trees have been employed to handle the vagueness in sensed data and to represent the level of uncertainty in the suggestions to the user. Si *et al.* [15] have developed a platform that can learn user behaviors from context history, in order to provide the most relevant services in the current situation. Here, Bayesian Networks are used to correlate contexts and services, by modeling the relationship between the sensor data and the selected services in the context history. Yap *et al.* [17] have proposed to dynamically choose the set of contextual information on which the resource recommendations can be based. Support Vector Machines techniques are applied to the context history in order to learn a relevance coefficient of each contextual information. Hong *et al.* [9] have suggested the use of context history to automatically extract the user preferences about services. In particular, by means of decision trees and association rules, the system is able to associate user context with services and even predict the next services the user might need.

In this paper, we first perform an accurate analysis phase to model the application domain and the corresponding context by ontologies and semantic rules, taking a generic user interacting with the mobile device into account. Then, we build a prototype of the recommender and supply the user with this prototype. During the use, we automatically collect some context history to extract the behavior of the user in the different situations of his/her typical day. This context history is used by a GA to adapt to the specific user behavior the meaning of the linguistic values used in the fuzzy linguistic rules of the resource recommender. This allows personalizing the resource recommender on the user behavior, thus improving its accuracy in timely suggesting the correct resources.

III. OVERALL ARCHITECTURE

The overall system architecture of the resource recommender is shown in Fig. 1. Here, we will illustrate only the main blocks of this architecture. The interested reader can find a more detailed description in [6], where we have also shown a comparison with other recently proposed recommenders. In the server side, the *semantic engine* and the *fuzzy engine* are the main modules. The semantic engine infers one or more current situations by exploiting domain knowledge modeled by ontologies (expressed in the Web Ontology Language – OWL [18]) and semantic rules (expressed in the Semantic Web Rule Language – SWRL [19]). Fig. 2 shows basic concepts and relationships identified for the situation ontology. In Fig. 3, we provide an example of semantic rule expressed in natural language by using the ontology of Fig. 2. In the rule, we have represented the conditions which typically are affected by a degree of uncertainty in italic bold. These conditions are modeled by using fuzzy propositions expressed in terms of linguistic variables and linguistic values in the *fuzzy engine*. These propositions are therefore connected by a logical AND implemented by using the minimum operator in order to form a fuzzy linguistic rule. Once fired, this rule can compute a certainty degree for the situation inferred by the corresponding semantic rule in the *semantic engine*.

The interoperability between the *fuzzy engine* and the *semantic engine* modules is guaranteed by the *observer*. More specifically, the *observer* module transmits to the *fuzzy engine* each contextual value which is affected by uncertainty. Then, the *fuzzy engine* checks whether the value belongs to a fuzzy set in the linguistic variable at some degree. If this occurs, the *observer* communicates to the *semantic engine* that the corresponding condition in the semantic rule can be considered true, thus triggering the semantic inference process. Obviously, the value can belong to more than one fuzzy set and therefore more conditions in different semantic rules are considered true, thus firing more than one semantic rule and possibly inferring more than one situation. Since the fuzzy engine computes a degree of certainty for each situation, taking the intrinsic vagueness of some conditions of the semantic rules into account, the system can associate a degree of certainty with each situation inferred by the *semantic engine*. Each situation is therefore associated with specific tasks on the basis of domain knowledge expressed in terms of a task ontology. Finally, the specific current task together with contextual information is used to recommend a set of resources, identified by means of a Label (or Tag)-based file system. The resources are recommended in the same order of the situations with which they are associated: from the resources associated with the situation characterized by the highest degree of certainty to the ones associated with the situation with the lowest degree of certainty.

The *application controller* module handles the execution flow of the server-side application, managing the activities of the other modules and acquiring data collected by the *contextual data sources* package. Contextual data concern geographical maps, user calendar, user position, user speed, Point Of Interests (POIs) for the user. In particular, the *application controller* drives the process of recording the acquired contextual data over the time to build the *context history* for the user.

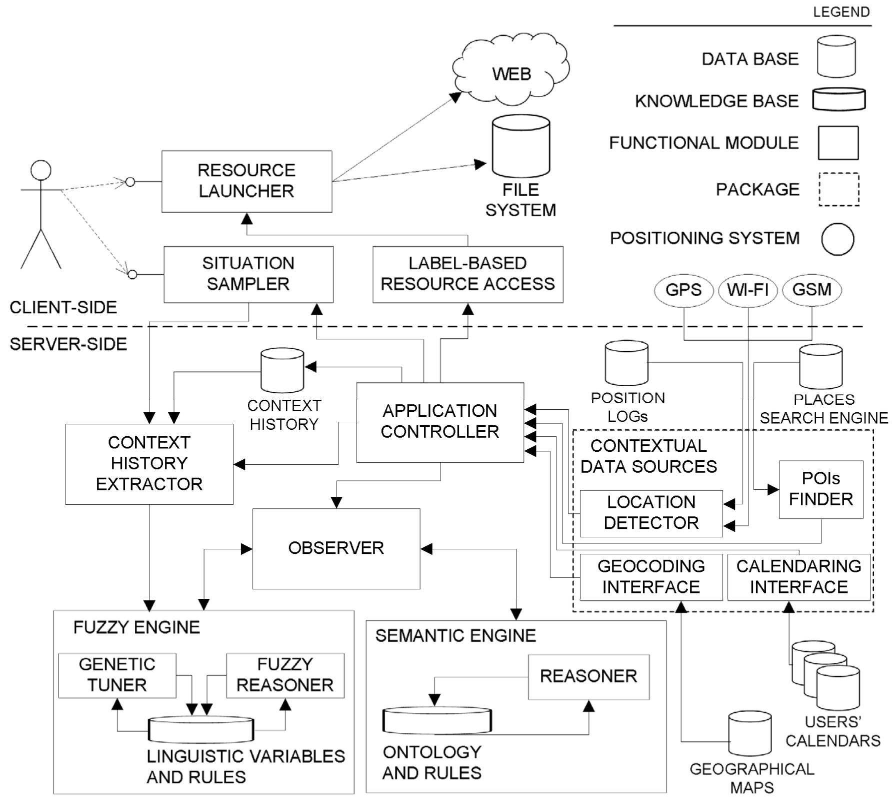

Figure 1. The overall system architecture.

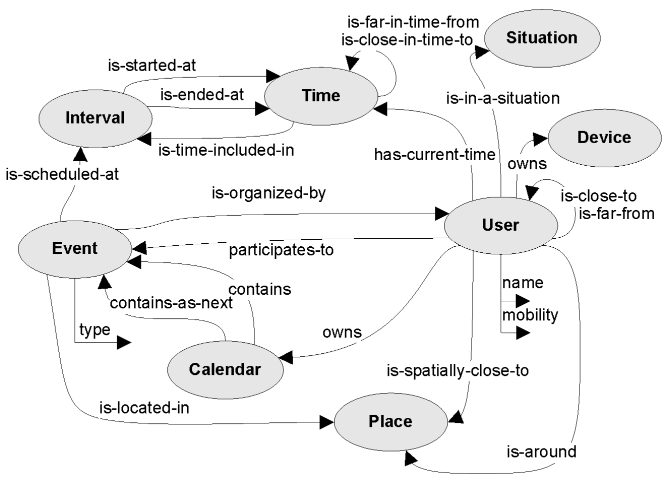

Figure 2. The situation ontology.

The *context history extractor* module is aimed at producing the training set needed for the tuning of the *linguistic values* so as to adapt the *fuzzy engine* to the specific user. In particular, the module associates a set of tracks of contextual data (context history) with the corresponding situations. In the *fuzzy engine*, the genetic tuner implements a GA that optimizes the membership functions associated with the linguistic values, as detailed in the next section.

On the client side, the *label-based resource access* [2] module provides a reference to tagged resources. Indeed, the application controller identifies the recommended resources by using abstract descriptors (labels) in place of their URIs, with the aim of being independent of the resources and enabling reusability of the ontology [5][6]. The reference to the resource is employed by the *resource launcher module*. Finally, the *situation sampler* module allows tracking the instants of time and the situations for the context history extractor, during the tuning phase. The tuning phase can be started manually by the user or automatically by the client application depending on a performance index that is monitored on the client side.

```
IF user1 PARTICIPATES-TO meeting1
AND meeting1 HAS-TYPE "business"
AND user1 HAS-MOBILITY "stationary"
AND user1Time IS-TIME-INCLUDED-IN meeting1Time
AND user1 IS-SPATIALLY-CLOSE-TO meeting1Place
THEN user1 IS-IN-A-SITUATION "ongoing-meeting"
```

Figure 3. An example of a semantic rule in natural language.

IV. The Genetic Tuner

Each rule in the fuzzy engine is expressed by using linguistic variables. For each linguistic variable, we define a set of linguistic values and associate a fuzzy set with each of these linguistic values. The fuzzy sets describe the meaning of the linguistic values. This meaning is generally fixed by considering a generic user. Actually, different users have different behaviors. Thus, it can be a very hard task to find a meaning which satisfies all the possible users. As an example, let us consider the rule shown in Fig. 3. To infer a degree of certainty for the situation "ongoing-meeting", the spatial closeness to the meeting place has to be evaluated. To this aim, a linguistic variable is defined with two linguistic values: *close* and *not-close*. To define a meaning for these two values for a generic user is not however a trivial task. Indeed, a very precise user would usually note in his calendar the complete address (street and number) of the meeting place, whereas a less precise user might note the street name only. To timely recognize the closeness of the user to the point of interest, the fuzzy set corresponding to the linguistic value *close* should be representative of both users. Considering the difference between the user habits, to achieve this objective is practically impossible. Indeed, a fuzzy set characterized by a narrow support would not allow to detect the closeness to the meeting place for the less precise user, whereas a fuzzy set with a wide support would detect too early the closeness for the very precise user.

To lessen this drawback and therefore improve the performance of the resource recommender, the specificity of each user has to be taken into account. This can be performed by employing the context history of the specific user for adapting the meaning of the linguistic values used in the rules of the fuzzy engine. To this aim, we can adopt a GA.

GAs have been so widely used to tune membership functions of linguistic values in fuzzy rule-based systems that a specific term, genetic fuzzy systems, has been coined in the literature [8]. Although in the last years different algorithms and procedures have been proposed to learn membership functions from data [10], in this paper, we adopt a very simple approach. On the other hand, our aim is only to show that the use of a tuning mechanism for adapting the resource recommender to the user habits and behaviors can considerably increase its accuracy and responsiveness.

Let us consider the generic linguistic variable X_j shown in Fig. 4. We assume that each linguistic value is represented by a trapezoidal membership function, $A_{j,t}$, whose support is $[a_{j,t}, d_{j,t}]$ and whose core is $[b_{j,t}, c_{j,t}]$. Further, for each fuzzy set $A_{j,t}$, $t=1\ldots T_j - 1$, we suppose that $c_{j,t} = a_{j,t+1}$ and $d_{j,t} = b_{j,t+1}$. Finally, $a_{j,1} = b_{j,1}$ and $c_{j,T_j} = d_{j,T_j}$ coincide with the left and right extremes of the universe, respectively. Thus, the strong partitions made of these membership functions can be represented by T_j - 1 pairs $(a_{j,t}, b_{j,t})$. Let M be the number of linguistic variables which have to be tuned. The overall data base can be defined by the chromosome shown in Fig. 5.

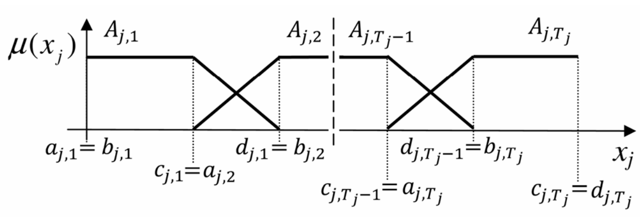

Figure 4. A generic partition of a linguistic variable.

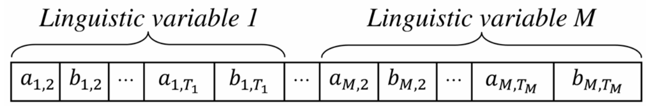

Figure 5. The chromosome coding.

We aim to tune the membership functions so as to increase the capability of the system to recognize the desired situation. To this aim, we maximize the following fitness function f. Let s_1, …, s_S be the possible situations the recommender system can recognize. Let s_t be the target situation. Then, f is defined as:

$$f = \sum_k \left(\mu_{s_t}(k) - \max_{r \neq t} \left(\mu_{s_r}(k) \right) \right) \tag{1}$$

where μ_{s_t} and μ_{s_r} are the certainty degrees with which the fuzzy engine recognizes the target situation s_t and each situation s_r different from s_t for the sample k in the training set. The training set is built on the basis of samples of the context history. Each sample is made of the contextual variables that allow inferring the situation, together with the user situation itself. To give a glimpse of the context history, let us consider again the semantic rule reported in Fig. 3.

Here, the context history is made of: (i) the mobility of the *user1*; (ii) the temporal inclusion of the *user1* time in the *meeting1* time; (iii) the spatial closeness of the *user1* position to the place of the *meeting1*. These contextual values are periodically recorded and associated to the respective user situations. Once the training set is large enough (a few hundreds of samples for each situation), the GA can be executed. We would like to highlight that for each observed situation, we store approximately an average of 400-450 samples (about one per minute).

The initial population of the GA is made of 50 chromosomes. Each individual of the population is randomly generated within the universe of the base variables. We adopt a BLX-α crossover operator with $\alpha = 0.5$ [7], an adaptive feasible mutation operator [16] and stochastic uniform selection [1]. The algorithm stops when the average fitness of the population, over 2000 generations, varies less than 10^{-6} . At the end of the GA execution, the membership function parameters are tuned by using the values of the chromosome with the highest fitness value.

V. EVALUATION CASE STUDY

We applied the proposed framework to two real business case studies, concerning a pharmaceutical consultant in typical business situations and an off-site student during a week of lessons [6]. For the sake of brevity, we will focus only on the first case study. On the other hand, the second case study is very similar in terms of methods and results.

More specifically, by means of a series of interviews, we extracted the following situations in which the consultant can be involved: (i) *Pre-Meeting on Movement*, when the user is moving towards the location of the upcoming meeting; (ii) *Ongoing-Meeting*, when the user is attending a meeting; (iii)

Meal, when the user is having lunch; (iv) *Post-Meeting on Movement,* when the user is returning home at the end of the work day. Hence, domain-specific ontologies and semantic rules have been developed accordingly.

In order to assess the effectiveness of the proposed personalization technique, a comparative analysis has been made with the performances achieved by a recommender configured by a domain expert.

To generate the training samples, a set of 10 tracks and 110 total events have been recorded in two working weeks of a pharmaceutical consultant.

The linguistic variables involved in the case study are: (i) *spatial closeness*, which represents the distance of the user from a place expressed linguistically as *close* and *not-close*; (ii) *temporal relativity*, which denotes the order between two instants of time and is expressed linguistically as *before* and *after*; (iii) *time inclusion*, which assesses whether an instant of time belongs to a temporal interval and is expressed linguistically as *included* and *not-included*; and (iv) *user mobility,* which represents the speed of the user and is expressed linguistically as *stationary* and *not-stationary.* Fig. 6 shows the linguistic variables defined by the domain expert and tuned by the GA, respectively.

After the tuning process, the two recommenders have been tested by a pharmaceutical consultant during a working week with 77 total new events. To asses the reliability and timeliness of the recommender, the following performance index has been considered. Let us assume that for each type E_i of event, we have N_i occurrences $o_{i,p}$. For each occurrence $o_{i,p}$, we record the instant of time $t_{i,p}$ at which that occurrence occurs, and the time $t'_{i,p}$ at which the recommender recognizes the occurrence. Let us define the *responsiveness* of the recommender to the type E_i of event as:

$$Resp(E_i) = \frac{\sum_{p=1}^{N_i} \left| t'_{i,p} - t_{i,p} \right|}{N_i} \qquad (2)$$

Table 1 shows the responsiveness of the two recommenders for each type of event occurred during the testing. Experimental results show that the recommender tuned by the GA considerably outperforms the recommender configured by the domain expert. In particular, with the proposed technique the recommender is able to increase the responsiveness on average of almost 30%, with peaks of 45% for some specific type of events.

To assess the capability of our approach to adapt the meaning of the linguistic terms to the user behavior, we have applied the GA to a different pharmaceutical consultant in similar business situations. Fig. 7 shows the linguistic variables after the GA optimization. By comparing Fig. 7 with Fig. 6.b, we can observe that the abscissas corresponding to the crossing points between *after* and *before*, and *included* and *not-included* in, respectively, linguistic variables *temporal relativity* and *time inclusion* are considerably smaller for the second consultant than for the first. This can be explained by the different habits of the two consultants. Indeed, the second consultant is typically latecomer whereas the first consultant is generally punctual.

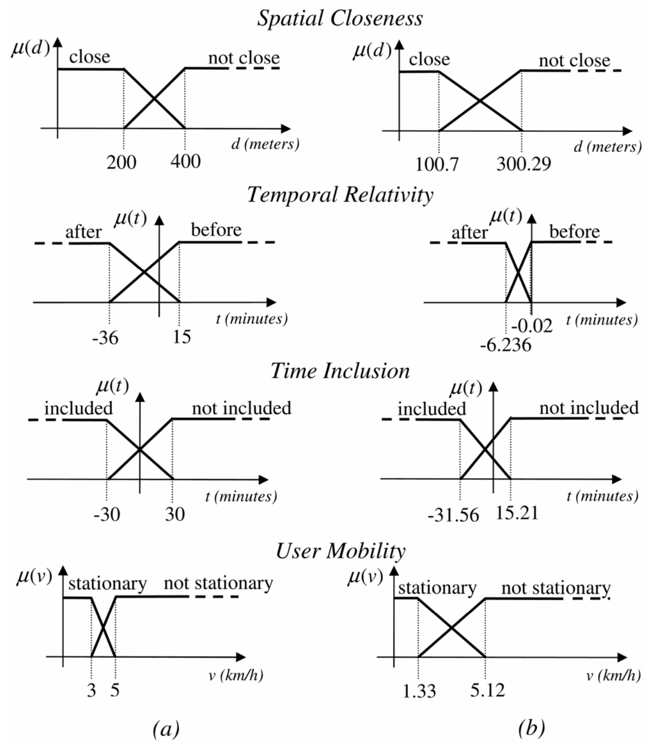

Figure 6. Linguistic variables for the evaluation case study defined by the domain expert (a) and tuned by the GA (b).

TABLE I. RESPONSIVENESS OF THE TWO RECOMMENDERS.

Events	**Responsiveness (in seconds)**	
	Recommender defined by the domain expert	***Recommender tuned by the GA***
Pre-meeting (begin)	49.71	35.71
Pre-meeting (end)	85.64	47.43
Ongoing Meeting (begin)	110.43	83.86
Ongoing Meeting (end)	31.93	22.93
Meal (begin)	101.29	68.86
Meal (end)	59.00	48.57
Post-meeting (begin)	27.33	23.00
Average	**66.48**	**47.19**

VI. CONCLUSIONS

In this paper, we have presented a method based on a GA to adapt a resource recommender to the behavior of the specific user. This allows to increase the accuracy of the recommender in determining the user situation, thus improving the effectiveness and reliability in suggesting the correct resources to the user. We have adopted a resource recommender proposed in our recent works. This recommender exploits fuzzy linguistic variables to manage the inherent vagueness of some contextual parameters. The GA tunes the meaning of these linguistic variables on the basis of context history collected by tracking the behavior of the user when interacting with the mobile device.

The effectiveness of the proposed method has been shown by testing the recommender in a real-world scenario. To this aim, a prototype has been implemented and configured for a case study, concerning pharmaceutical consultants in typical business situations. The experimental results have shown that the GA enhances the performance of the recommender, increasing its responsiveness and modeling capabilities.

Ongoing works concern the adaptation of the recommender to user habits that can occasionally change.

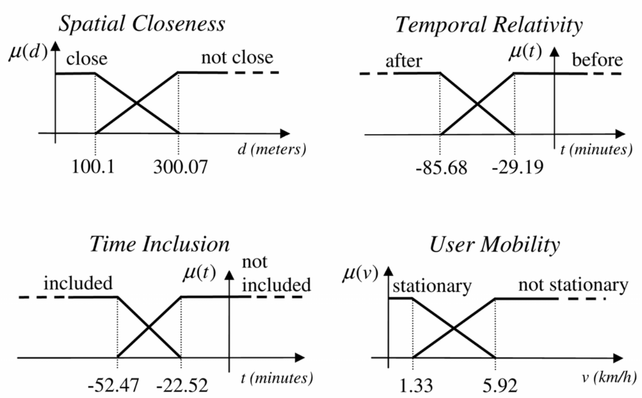

Figure 7. Linguistic variables tuned by the GA on the behavior of another pharmaceutical consultant.

REFERENCES

[1] J.E. Baker, Reducing bias and inefficiency in the selection algorithm in Proc. of the Second International Conference on Genetic Algorithms and their application, pp. 14-21, 1987.

[2] S. Bloehdorn, O. Görlitz, S. Schenk, and M. Völkel, "TagFS, tag semantics for hierarchical file systems", in Proc. Of the 6th International Conference on Knowledge Management, (IKNOW'06), pp. 6-8, 2006.

[3] H.E. Byun and K. Cheverst, "Supporting proactive intelligent behaviour: the problem of uncertainty", in Proc. of the Workshop on User Modeling for Ubiquitous Computing (UM03), Johnstown, PA, pp 17-25, 2003.

[4] H.E. Byun, and K. Cheverst, "Utilizing Context History to Provide Dynamic Adaptations, Applied Artificial Intelligence", Vol. 18, No 6, pp. 533-548, 2004.

[5] A. Ciaramella, M.G.C.A. Cimino, B. Lazzerini, and F. Marcelloni, "Situation-aware mobile service recommendation with fuzzy logic and semantic web", in Proc. IEEE International Conference on Intelligent Systems Design and Applications (ISDA'09), pp. 1037–1042, 2009.

[6] A. Ciaramella, M.G.C.A. Cimino, B. Lazzerini, and F. Marcelloni, "A situation-aware resource recommender based on fuzzy and semantic web rules", International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems, Vol. 18, No. 4, pp. 411-430, 2010.

[7] L.J. Eshelman, and J.D. Schaffer, "Real-Coded Genetic Algorithms and Interval-Schemata", Foundations of Genetic Algorithms 2, Morgan Kaufman Publishers, San Mateo, pp. 187-202, 1993.

[8] F. Herrera, "Genetic Fuzzy Systems: Taxonomy, Current Research Trends and Prospects," Evolutionary Intelligence, Springer-Verlag, vol. 1, pp. 27–46, 2008.

[9] J. Hong, E.-H. Suh, J. Kim, and S.Y. Kim, "Context-aware system for proactive personalized service based on context history", Expert Systems with Applications, Elsevier, Vol. 36, No 4, pp. 7448-7457, 2009.

[10] M. Kaya, and R. Alhajj, "Genetic algorithm based framework for mining fuzzy association rules", Fuzzy Sets and Systems, Vol. 152, No 3, pp. 587-601, 2005.

[11] W.P. Lee, "Deploying personalized mobile services in an agent-based environment", Expert Systems with Applications, Vol. 32, No. 4, pp. 1194–1207, 2007.

[12] M. Luther, Y. Fukazawa, M. Wagner, A. Kurakake, "Situational reasoning for task-oriented mobile service recommendation", The Knowledge Engineering Review, Cambridge University Press, Vol. 23, pp. 7-19, 2008.

[13] R. Mayrhofer, "An architecture for context prediction", Trauner Verlag, Schriften der Johannes-Kepler-Universität Linz, volume C45, April 2005.

[14] C. Porcel, A.G. Lopez-Herrera, and E. Herrera-Viedma, "A recommender system for research resources based on fuzzy linguistic modeling", Expert Systems with Applications, Vol. 36, No 3, Part 1, pp. 5173-5183, 2009.

[15] H. Si, Y. Kawahara, H. Morikawa, and T. Aoyama, "Stochastic approach for creating context-aware services based on context histories in smart home", in Proc. of exploiting context histories in smart environments (ECHISE '05), pp. 3480–3495, 2005.

[16] P. Vasant and N. Barsoum, Hybrid genetic algorithms and line search method for industrial production planning with non-linear fitness function, Engineering Applications of Artificial Intelligence, Elsevier, Vol. 22, No 4-5, pp. 767-777, 2009.

[17] G.-E. Yap, A.-H. Tan, and H.-H. Pang, "Dynamically-optimized context in recommender systems", in Proc. of the 6th International Conference on Mobile Data Management (MDM'05). ACM Press, New York, NY, USA, pp. 265-272, 2005.

[18] W3C, OWL Web Ontology Language Reference, W3C Recommendation, 10 February 2004, electronically available at http://www.w3.org/TR/owl-ref/.

[19] W3C, SWRL: A Semantic Web Rule Language Combining OWL and RuleML, W3C Member Submission 21 May 2004, electronically available at http://www.w3.org/Submission/SWRL/.