

Document downloaded from:

<http://hdl.handle.net/10251/35311>

This paper must be cited as:

Mocholi Agües, JA.; Jaén Martínez, FJ.; Krynicki, KK.; Catalá Bolós, A.; Picón, A.; Cadenas, A. (2012). Learning semantically-annotated routes for context-aware recommendations on map navigation systems. *Applied Soft Computing*. 12(9):3088-3098. doi:10.1016/j.asoc.2012.05.010.



The final publication is available at

<http://dx.doi.org/10.1016/j.asoc.2012.05.010>

Copyright Elsevier

Learning semantically-annotated routes for context-aware recommendations on map navigation systems

Jose A. Mocholi, Javier Jaen, Kamil Krynicki, Alejandro Catala, Artzai Picón^a, Alejandro Cadenas^b

ISSI - Departamento de Sistemas Informáticos y Computación, Universidad Politécnica de Valencia,
Camino de Vera, s/n, 46022 Valencia, SPAIN
{jmocholi, fjaen, kkrynicki, acatala}@dsic.upv.es

^aTECNALIA-Infotech, Parque Tecnológico, Edificio 202, 48170 Zamudio, Bizkaia, SPAIN
artzai.picon@tecnalia.com

^bTelefónica I+D, Emilio Vargas, 6, 28043 Madrid, SPAIN
cadenas@tid.es

Corresponding author:

Jose A. Mocholi
Camino de Vera, s/n, 46022 Valencia, Spain
Email: jmocholi@dsic.upv.es
Phone: +34 96 387 35 69
Fax: +34 96 387 73 59

ABSTRACT

Modern technology has brought many changes to our everyday lives. Our need to be in constant touch with others has been met with the cellphone, which has become our companion and the convergence point of many technological advances. The combination of capabilities such as browsing the Internet and GPS reception has multiplied the services and applications based on the current location of the user. However, providing the user with these services has certain drawbacks. Although map navigation systems are the most meaningful way of displaying this information, the user still has to manually set up the filter in order to obtain a non-bloated visualization of the map and the available services. To tackle this problem, we present here a semantic multicriteria ant colony algorithm capable of learning the user's routes, including associated context information, and then predicting the most likely route a user is following, given his current location and context data. This knowledge could then be used as the basis for offering services related to his current (or most likely future) context data close to the path he is following. Our experimental results show that our algorithm is capable of obtaining consistent solutions sets even when multiple objective ontological terms are included in the process.

Keywords

Semantic Search, Context-awareness, Ant Colony Optimization, Ontologies.

1. Introduction

Advances in technology have changed our lives in many ways. First, miniaturization of electronics made it possible to put in our pockets voice recorders, cameras, GPS receivers and other electronic devices with some computing capabilities such as electronic agendas. The spread of cellphone networks allowed us to be in touch with others at all times. Later, as more breakthroughs became mature and mainstream, devices combining multiple capabilities appeared and, given our wish to be permanently communicable, cellphones were the point of convergence for many of these capabilities. Current cellphones can now carry out more tasks than simply communicating, such as browsing the Internet, playing multimedia files, showing maps or taking photos, among others.

People want to be connected while on the move. More and more services and applications make use of current user location (by means of a GPS antenna) to deliver a more tailored experience or more meaningful information to the end user. For example, in [1] raw GPS data is mined to determine which transport mode (driving, walking, etc.) is being used, so that users can understand better their own life patterns and easily share this information with others. In this work context-aware computing, based on estimating the user's present transport mode, is enabled and application systems are empowered to distinguish GPS tracks by transport modes to help users to find the best route to their destinations. In general, these *location-based services* (LBS) are described as services that are accessible by means of mobile devices connected to a mobile network and provide location-based information supplied by the mobile device [2]. Other examples of LBS applications include roadside assistance, fleet management, map navigation, weather information and personal location services.

As the number of LBS applications increases, map navigation systems have become a very effective system for displaying all kinds of location-based information. However, contextual information (i.e., data related to a certain context, such as location, time, activity or temperature, and that may be used to represent it) is not generally used to improve the visualization of information on these systems. Four information generations can be distinguished, according to how this information is shown on map navigation systems:

- 1st generation (*non context-driven navigation* systems) consist of map navigation systems that do not have the capacity to detect and maintain the user's context, such as his current location, the user is thus responsible for providing his context data and keeping it in mind while he is manually navigating by the map display.
- 2nd generation (*location-based* navigation systems) are capable of detecting the user's current location and displaying it on a map by means of a GPS antenna, gyroscopes, accelerometers, or WIFI/3G signal triangulation. The user's location is the only context data that is automatically detected and the system displays nearby *points of interest* (POI) such as museums, hospitals, hotels, airports, gas stations, or any location-based service.
- The distinguishing characteristic of 3rd generation map navigation systems is that they filter out some of the available POIs by using context information entered manually by the user. This context information can be expressed in terms of preferred POI categories, within a

certain distance from his current location, keywords that semantically describe preferred POIs or other contextual data. Almost all modern mainstream map navigation systems belong to this generation.

- 4th generation (non-temporal context-driven navigation systems) are capable of learning the user's preferences and inferring the present context to tailor the user experience [3]. However, these systems may not offer information about areas the user is likely to be traversing in the near future anticipating his (her) needs for information and services.

Despite all the features navigation systems have incorporated over time, we consider there is a need for one that could effectively improve the way information is displayed. If contextual information of the user's routes could be combined with their spatial coordinates, then a semantic model of the journeys could be established. This route-driven semantic model could then be used to predict the most likely route(s) and/or destination(s) from his present position, by semantic matching algorithms, offering useful information tailored to his contextual situation. Such a system could be considered the 5th generation of map navigation systems and would allow, for example, drivers to be aware of traffic problems ahead. Another example would be reminding a user of an important meeting on the way to work, or the need to buy milk on the return trip home. They could also be used for location-based advertising, as users could receive advance information on discounts or sales in the vicinity of the expected route, when users can modify their trips. Finally, these semantic models can also be used to design routes for new users (when no model has been learnt in advance) based on the models of other users in the vicinity at the same time or in the past. This sharing of semantic information enables interesting scenarios, such as being able to offer tourist information (with no previously learnt information) on the expected route by analyzing the semantic models previously learnt from other users using the same mode of transport.

In this paper we propose an *ant colony optimization* (ACO) algorithm that is capable of collecting contextual data related to journeys made by users, learning the sequences of this contextual information, predicting the most likely future contextual sequences based on the current context of the user and displaying only the most appropriate data (e.g. POIs, to-do list items, traffic problems, etc) taking into account the spatial and semantic proximity to the predicted route(s). The paper is organized as follows: in Section 2 we review similar studies. Section 3 gives an overview of ACO, provides a general definition of the problem and describes the functionality of the proposed algorithm. Section 4 presents the experiment conducted to test the algorithm, explains the terms of the ontology used and discusses the results obtained from the execution of the algorithm with different sets of criteria. Besides, this section also presents the results of an empirical comparison between two well known search algorithms that have been widely used in the literature and the one proposed in this paper. In Section 5 we present our conclusions and describe future work.

2. Related studies

Despite the number of features that map navigation systems have incorporated over time, most mainstream commercial systems still lack a crucial characteristic for systems of the 5th generation, namely the ability to learn semantic information from past trips to enhance user experience and offer better results in the future. An example of this is the IQ Routes feature found in some TomTom products [4], which

uses anonymous trip data from its huge worldwide customer base to calculate the best routes at the time, based on average speed. However, the user still has to manually enter his destination and, due to its anonymous nature, better use could be made of the user's contextual information. In fact this system only needs a predefined ontology (set of variables to be measured), as the minimization function (fastest route) is known in advance and hardcoded as part of the software.

However, this does not mean nothing has been done on the creation of systems that would fit in what we have presented as the 5th generation of map navigation systems. In [5] the authors use records of an individual's location over an interval of time (location histories), clustering algorithms to identify destinations and a Markov Model to model transitions between clustered locations. This allows distinguishing between typical and atypical patterns of behaviour and predicting future user locations from their recent location histories. In [6] Hidden Markov Models (HMM) are used to capture a sequence of driving actions to deduce the destination, based on a present and next state transition model. In [7] some algorithms were developed to extract relevant places from GPS data, which were then used to train Markov models (one for each location, with transitions to every other location) with the aim of predicting the next or most likely destination. The algorithms developed to identify significant places from GPS data were able to find common relevant places from several users, which means that if two people A and B have some places in common and there is not enough data from A to build a successful predictive model, B's collection of places and his predictive model may be used as the starting point of A's predictive model. In [8], data is provided by other users to improve the results for a certain user and his history of destinations plus those of others is used to predict the destination (not the complete route, which is our aim) as his trip progresses. It divides a map in cells and uses a Bayes formula to calculate the probability that a cell will be the destination, based on which cells the driver has already visited. This work is focused on car drivers and the authors suggest that its use to proactively deliver information (upcoming POIs or traffic problems) could reduce the cognitive load on the driver, as the information about unlikely waypoints is discarded. In the comMotion study [9] a hybrid approach is followed and histogram matching, a HMM and a Bayes classifier are used to predict the user's destination. It is able to learn routes, predict the route that is being followed and, hence, the destination. With this information, as in [8], comMotion provides just-in time information delivery, such as reminding users of an important meeting on the way to work, or the need to buy something on the trip home. A similar goal is pursued in [10] where similarity and clustering algorithms are used to predict a vehicle's route in order to give drivers warnings about upcoming traffic hazards or information about upcoming POIs. Another system with just-in time information delivery in mind is presented in [11], where system administrators can configure the delivery of alerts, advertisements, or any other information based on system policies as well as user profiles and preferences. In this work, spatial databases are used to store the path records of all users. This allows the path prediction algorithm to transit between an individual user path history and all the stored path records in case a certain user does not have a sufficient path history.

The study [12] makes use of more context data apart from GPS coordinates and the authors use a Bayesian network hierarchy to learn personal maps customized for each user. A personal map includes personally significant places and routes, which allows the system to recognize significant locations for a user and the activities associated with those places, infer transportation modes and destinations, and

detect user errors or novel behaviour. In [13] a context data logging software installed on a cellphone is used to log several context variables (such as GPS data, current GSM cell ID, phone profile, active phone application, etc.) and an algorithm is proposed to learn important locations and routes between a pair of locations, and to predict likely routes and destinations. The system then determines traffic conditions along the predicted routes and delivers timely advisories and alerts to the user when there are unexpected slowdowns.

Although several of the studies presented above include the delivery of information among their goals, some of them do not make route predictions, which makes the provision and filtering of location-based information more difficult. Most of them do not use context data other than location and, if they do, it is not very meaningful for the users (for example, the GSM cell ID, the list of Bluetooth devices around the phone, the battery status or the phone idle/active time do not supply any useful context information for most users). We therefore propose in this paper an algorithm based on the ACO metaheuristic that is capable of predicting most likely routes and destinations for the current location and contextual data of the user, based on previously learnt routes and their associated context data. The algorithm, named *SACO* (Semantic Ant Colony Optimization), can operate either in automatic or on-demand modes. In automatic mode the algorithm uses only the current user location and contextual data to make predictions and display information for the most likely routes and destinations, while in on-demand mode, users can ask for a set of likely routes for several given context data values. Another feature of our proposal is that it is not forced to use a prefixed set of context variables, allowing users to define individual meaningful context variables that the algorithm will use in the form of an ontology. Finally, as observed in our previous studies [14] [15] [16], the ACO has a design that can be easily parallelized, since the colonies can be seen as individual computational nodes. This allows finding implementations that can be parallelized to serve larger user models or a higher number of users while achieving a good trade-off between scalability and performance. In this sense, the comparison we present in our experimental section of our algorithm against other effective techniques for searching in graph-based problems will prove that the selection of ACO is the right choice for scalability while preserving search effectiveness.

Table 1 summarizes the characteristics of the related works and *SACO* to compare their respective capabilities. It may be observed that the works are classified into those that try to predict the user destination only and those that try to predict the complete route. Among the later ones, only the system proposed in this paper and the work in [13] returns a set with the most probable routes the user is following. Although there are two related works that use other context data apart from GPS coordinates, just *SACO* is generic to allow using any ontology to describe context in a certain domain. Besides, only a subset of the related works is capable of learning, i.e. of adding new knowledge about the user, while they are operating. However, the learning process is either restricted to a fixed predefined ontology or it only allows learning contextual situations as data related to a final destination. Finally, *SACO* is the only work capable of performing a semantic search of the likeliest routes given a certain context, that is, any of the terms in the ontology may be used to search for routes in the vicinity of the semantic space (minimizing their semantic distance with respect to the terms included in the semantic query) as it will be discussed in the following sections.

3. Semantic ACO algorithm for context-aware recommendations

3.1 Ant Colony Optimization overview

Ant colonies are insect societies that accomplish complex tasks by presenting highly structured organizations and communication mechanisms. Ants' behaviour is based on the use of pheromones, including the trail pheromone, which is particularly interesting since it is used to mark paths on the ground from food sources to the nest. Several experiments [17] have shown that this communication mechanism is very effective in finding the shortest paths and has thus inspired several stochastic models [17] that imitate the dynamics of these colonies. Based on this natural behaviour and the proposed stochastic models, Dorigo and his colleagues [18] first introduced Ant Colony Optimization (ACO) as a metaheuristic that targets combinatorial optimization problems, where the problem search space is generally represented as a graph. ACO algorithms make use of artificial ants, which are stochastic constructive procedures that build step-by-step solutions following a decision process that is based on shared information on pheromone trails. ACO algorithms have been used to solve routing, assignment, scheduling, subset, machine learning and routing problems [19].

In ACO, m independent ants obtain m different solutions (tours) by iteratively choosing a node from the graph as the next node to be appended to their respective tour. The selection of the next node to visit is carried out either by randomly exploring the search space or by applying a probabilistic state transition rule that takes into account the attractiveness of the nodes and the pheromone trails that have been deposited by other artificial ants (see [20] for a mathematical definition of this process). Once m solutions have been obtained, the best tour is selected, pheromone trails are intensified for the graph edges in the winning tour and evaporation is applied to the remaining trails in the graph. It has been demonstrated that, by iteratively proceeding in such a way, the algorithm converges to suboptimal solutions [19]. In terms of our specific problem domain, ants probabilistically select the next location to add to the route by using both pheromone information and the local-heuristic of nodes (the score assigned to a certain location is calculated from the similarity between its context data and the current/provided context data). The generic ACO algorithm can be summarized in a pseudo-code schema as follows:

```
Set all parameters and initialize pheromone trails
Loop
  Sub-Loop
    Construct solutions based on the state transition rule
    Apply the online pheromone update rule (optional)
  Continue until all ants have been generated
  Apply local search (optional)
  Evaluate all solutions and record the best solution so far
  Apply the offline pheromone update rule
Continue until the stopping criterion is reached
```

The algorithm includes the possibility of using some *local search* method with the goal of an improvement of the found solution. In [21] Dorigo and Gambardella elaborate on the use of local search methods (also known as local optimization heuristics) as a mechanism to refine solutions found by heuristics such as ACO. However, in [15] we performed a series of experiments to evaluate the effectiveness of these methods for the orienteering problem and concluded that the application of these methods contributes to the improvement of the quality of the solutions, but also incurs in significant computational costs.

In mathematical terms, the probabilistic selection rule (probability that ant k selects node j as the next one to be visited with the current position at node i) applies either exploitation by selecting the best node to be added to the tour, the node that maximizes attractiveness and pheromone level as shown in formula (1), or exploration by selecting nodes randomly according to a probability distribution function shown in (2). Whether the selection rule performs exploration or exploitation is controlled with a random number q that is generated uniformly between 0 and 1 and a threshold q_0 , as can be observed as follows:

$$\text{if } q \leq q_0$$

$$P_{ij}^k = \begin{cases} 1, & \text{if } j = \arg \max_{u \in \mathfrak{A}_i^k} \{[\tau_{iu}]^\alpha \cdot [\eta_{iu}]^\beta\} \\ 0, & \text{otherwise} \end{cases} \quad (1)$$

else

$$P_{ij}^k = \begin{cases} \frac{[\tau_{ij}]^\alpha \cdot [\eta_{ij}]^\beta}{\sum_{u \in \mathfrak{A}_i^k} [\tau_{iu}]^\alpha \cdot [\eta_{iu}]^\beta}, & \text{if } j \in \mathfrak{A}_i^k \\ 0, & \text{otherwise} \end{cases} \quad (2)$$

where η_{ij} is a problem-specific local heuristic, τ_{ij} are the pheromone trails, \mathfrak{A}_i^k the set of unvisited nodes and α, β are parameters that control the relative influence of the pheromone and local heuristic values respectively.

The pheromone update rule consists of two steps, an online and an offline updating. The goal of the online updating step is to decrease the level of pheromone of the selected move and make it less attractive for other ants, so new exploration is encouraged. It is performed by (3)

where $\varphi \in (0, 1]$ is a parameter that controls pheromone persistence and τ_0 is the initial pheromone trail intensity.

$$\tau_{ij}^{new} \leftarrow (1 - \varphi) \cdot \tau_{ij}^{old} + \varphi \cdot \tau_0 : \varphi \in (0, 1] \quad (3)$$

The offline updating step follows an elitist approach and only the arcs (i, j) of the best feasible solution found so far (S_{BEST}) are updated as follows:

$$\tau_{ij}^{new} \leftarrow (1 - \varphi) \cdot \tau_{ij}^{old} + \varphi \cdot \tau_{ij, S_{BEST}} : \varphi \in (0, 1] \quad (4)$$

3.2 SACO: An ontology-based multicriteria ACO algorithm

Our approach, denoted here as *SACO* (Semantic Ant Colony Optimization), is based on previous work [22] [23]. It carries out searches on route databases by utilizing a *domain-specific ontology* and the comparison between queries and learnt routes via conceptual distance measures [24]. Ontologies are a specification of an abstract view of the world that is represented for a specific purpose [25] and, therefore, define a set of representational terms called *concepts* which are interrelated to describe a target world. The SACO algorithm may be formulated as follows and its process is illustrated in Figure 1:

```
SACO

  Load ontology

  Read restriction set

  Apply restriction set (select and create nodes for the search space according to restriction set terms)

  Perform nodes score assignment process

  Apply ACO-OP algorithm

END SACO
```

SACO is different from other ACO algorithms, such as ACO-OP [20], in that the former uses semantic descriptions of concepts to dynamically assign scores to nodes representing problem domain entities in a process called *semantic score assignment*. In addition, this process is not performed statically and only once for every user according to his profile, but instead is a dynamic process that considers the restrictions defined either automatically or by the user. Nodes represent problem domain entities through the values of the ontology terms they contain. In the case of the problem described here, nodes represent a certain contextual situation and edges (the links between two nodes in the graph representing the problem search space) establish a sequentiality relation between two contextual situations. A sequential collection of contextual situations is called a contextual sequence.

Restrictions sets are defined in our approach by means of assigning relevance factors to elements in the ontological space. In terms of the problem presented in this paper, a relevance factor is assigned to a context data value, i.e. an ontology term value. The goal of the relevance factor is to adjust the level of importance of certain ontology term values over the rest within the ontological space.

Search space creation

All the ontology terms in the restriction set are used to obtain the search space and create the graph and its nodes. Thus, given a restriction set R that may be represented as follows:

$$R = \{ (term_1, value_1, r_1), (term_2, value_2, r_2), \dots, (term_n, value_n, r_n) \}$$

In order to create the nodes, we take the set of terms $\{term_1, term_2, \dots, term_n\}$ of the restriction set and select only those contextual sequences from the problem search space that have at least one of the restriction set terms among the metadata of their contextual

situations. For each one of these selected items our algorithm will create a graph node that will have, as associated metadata, all the pairs of $(term, value)$ for every term appearing in the restriction set.

As a result of applying a restriction set on the items of a collection that has been annotated with the terms of a given ontology, we obtain a graph that represents the search space to be used by our SACO algorithm. In this way, our approach can ensure both that the generated solutions will closely match the preferences specified by the user and good performance by reducing the problem search space, since restrictions help us to select the items which become relevant graph nodes.

Semantic score assignment

Based on relevance factors and restrictions sets, the semantic score assignment process assigns scores to each node in the search space as follows:

- Any item in the collection is defined as a tuple (ID_i, O_i) where ID_i is a unique identifier, and O_i is a set of pairs $\{(term_k, value_k) \mid term_k \in T, value_k \in Domain(term_k)\}$ representing values associated to terms in a set T of a predefined ontology.
- A user restriction set R is defined as a set where

$$R = \{(term_j, value_j, relevance_j) \mid term_j \in T, value_j \in Domain(term_j), relevance_j \in [0,100]\}.$$
- A distance metric over a term $term_k \in T$ is defined as a function $d_{term_k} : Domain(term_k) \times Domain(term_k) \rightarrow [1,100]$
- Given a user restriction set R , a collection of distance metrics over T , and given the set M of matching pairs $M = \{(term_j, value_j) \in R \mid \exists (term_j, value_j) \in O_i\}$, the score S_i associated to a node of the graph representing an item (ID_i, O_i) of the collection is calculated as follows:

$$S_i = \frac{\sum_{j=1}^{|M|} d_{term_j}(value_j, value_i) \cdot relevance_j}{|M|} \quad (5)$$

In the above formalization, if M is the empty set, then $S_i=1$, so that the transition probability function defined in (2) can be correctly obtained. Note that the score is proportional to the conceptual distance between the terms expressed in the query and the metadata description in O_i . In the current implementation, the collection of distance functions is usually obtained by the manual intervention of an expert user, who is able either to establish a conceptual fixed distance between different elements in the ontology or to create an evaluation function to calculate the distance between two values of a term of the ontology. Given this definition for obtaining the score associated to each node in the problem space, the ACO local heuristic when applying exploitation is as follows: $\eta_{ij} = S_j / c_{ij}$, where cost is calculated from the spatial distance between the coordinates metadata terms of the linked nodes. This means that nodes with a better ratio score/cost are more attractive to ants when exploiting pheromone information and will be selected during the solution building process. From a more general point of view, by controlling how η_{ij} is calculated (i.e., how score and cost are calculated) and the pheromone update rule, one can lead an ACO algorithm to look for those paths that are the best solutions for the kind of problem to be solved. For example, ACO

algorithms have been used to find the shortest path for the travel salesman problem [21], the path that visits the highest number of interesting artworks given a maximum time to visit a museum [15] or the critical path in project scheduling [26] among others.

SACO conceptual model

The conceptual model underlying our approach can be seen in Figure 2, which also includes the concepts defined above. The kind of available concepts (i.e. context data variables) defined by the ontology in use (an instance of the Ontology class) are represented by the Term class, and each one of its values is represented by objects of the Metadata class, which are also used to build up the RestrictionSet specified by the user in order to establish the characteristics of the solution(s) he wants to obtain.

Each Metadata instance in the RestrictionSet object has an associated Relevance factor, as explained previously. The algorithm works on Node instances and their contents are represented with Metadata values. There are 3 kinds of Node:

- The first type is represented by the SequenceNode class and represents a set of captured contextual data (e.g. location, time, activity, temperature, etc). As context data is captured at a certain pace, each new SequenceNode is linked to the previous one building up a ContextualSequence object. In the specific problem domain presented in this paper, an instance of the ContextualSequence class may be seen as a single trip with all its associated contextual data.
- The second one is the MetaNode and it consists of all the metadata values that are shared by all the nodes in a contextual sequence. Therefore, it is also used to represent learnt contextual sequences.
- The third type is represented by the ModelNode class, which instances represent nodes of the learnt contextual sequences. ModelNodes are linked to other ModelNodes in order to maintain the representation of the learnt contextual sequence.

The need for 3 types of nodes is explained by the process followed to “learn” a contextual sequence. Once a contextual sequence has been captured, it is added to the user model maintained by the system and represented by the Model class. Each user model holds the contextual sequences learnt through time as the user travels around and, since he generally follows the same routes with minimal differences, it is necessary to determine if a new contextual sequence about to be added is already present in the user model. Here the MetaNode class, the second Node type, comes into play. The merging process of a new contextual sequence starts by determining whether any of the existing MetaNode represents the sequence to be added. If no such a MetaNode is found, the system considers the sequence to be a new one to be appended to the user model and, consequently, the merging process creates a new learnt contextual sequence represented by a new MetaNode object and each SequenceNode of the contextual sequence to be added is converted to a ModelNode instance, the third type of Node. As model nodes are created and linked to existing ones, the semantic distance between them is used to calculate the cost of the edge that links them.

On the other hand, if there is one MetaNode whose contextual values are similar to the new contextual sequence, then the merging process will integrate (merge) the new sequence in the Model of the user. The merging process analyzes each SequenceNode in the contextual sequence to verify if there is a ModelNode with similar metadata values (within a certain threshold). If such a ModelNode is found, its metadata values are updated accordingly using a statistical approach so that it may be seen as the centroid of a cluster of sequence nodes

that share similar metadata values. If a sequence node is very different to its corresponding model node (e.g. a deviation from the usual route), the merging process will create a new model node. A model node may therefore be linked to more than one previous or following model nodes. This mechanism allows the representation of deviations in the usual route. The process of adding a contextual sequence to a user model is illustrated in Figure 3.

4. Experimental Results

The experiment described here was designed to evaluate the execution of the SACO algorithm. The restriction sets used to query the user model were thus manually established and supplied to the algorithm, although a software agent would be responsible for supplying the restriction sets and the context data used to build up and update the user model in a real case. This agent would also personalize the information to be visualized according to the user's current context.

The user model used in this experiment was created from approximately 120 different trips, which resulted in about fifteen different semantically annotated learnt routes because of the algorithm's merging process. Some of these routes were those customarily used and thus were learnt from several trips, but there are also some routes that were only used once. As trips were captured in the form of contextual sequences, the following contextual data were collected per node and, hence, they represent the terms of the ontology to be used:

- Activity: the kind of activity the user was involved in when the data was captured (e.g. work or leisure time).
- Time: the exact time of day when the data was captured.
- GPS coordinates: the user's location when the contextual data was gathered.

In order to interact with the algorithm and visualize the results of its execution, a desktop application was developed to display returned matching routes on a map. For this task, the application developed using the .Net framework makes use of a Google Earth plug-in, which allows loading a user model and creating a restriction set by adding ontology terms along with a relevance factor, as can be seen in Figure 4. It also allows a small amount of personalization for the visualization of the returned routes, by specifying the number of routes to be displayed and the number of intermediate model nodes per route to be shown on the map. The application was run on a Pentium4 2.4GHz processor with 1GB of RAM and the Windows XP operating system.

In Figure 5 it can be seen how the routes obtained from the algorithm for a given restriction set are displayed on the map.

As part of the experiment, several functions were defined to calculate the semantic distances between the values of every term in the ontology. These functions are used in the dynamic score assignment process and also to calculate the cost of the edge that links two model nodes, as previously mentioned.

In the rest of this section we will show the restriction sets used to query the user model and test the algorithm. A list with the routes obtained will be given for each one, ordered from highest to lowest score. The semantic distance from the restriction set is shown for each route in the list. The percentage of routes in the model with lower semantic distances than the worst obtained solution not selected by the algorithm as candidate solutions was another quality metric evaluated for each restriction set, in order to find out whether there were good

solutions that had been left out by our implementation of the algorithm. The closer this percentage is to 0 the better the algorithm's behaviour. Although it might seem that this metric should always be 0, the stochastic nature of the node selection process, together with a high number of bifurcations in the problem space, can make the algorithm obtain solutions that are local minima. To minimize this effect, the algorithm uses several ants and performs several iterations per ant. Despite this extra-processing work, it took about 5 seconds on average to get the results for each of the tested restriction sets.

Experiment 1: Restriction set: {(activity = work, relevance = 80), (time = 8.00, relevance = 80)}

	Activity	Time	Semantic distance	Duration
Route 1	work	8:00:02	5.564	14 min
Route 2	work	8:08:21	19.044	30 min
Route 3	work	8:01:21	21.864	48 min
Route 4	work	8:13:41	23.674	34 min
Route 5	work	8:05:58	30.375	1h 1min

The value of the percentage of routes with a semantic distance lower than 30.375 was 0 for this restriction set although there were some other routes that matched it but with higher semantic distances.

Experiment 2: Restriction set: {(activity = leisure, relevance = 80), (time = 22.00, relevance = 80)}

	Activity	Time	Semantic distance	Duration
Route 1	leisure	22:20:07	25.275	22 min
Route 2	leisure	18:06:41	100.694	4h 57 min
Route 3	leisure	19:09:13	120.738	40 min
Route 4	leisure	17:27:41	160.899	1h 33 min
Route 5	leisure	16:18:34	165.011	3h 16min

The value of the percentage of routes with a semantic distance lower than 165.011 was 0 for this restriction set although there were some other routes that matched it but with higher semantic distances.

Experiment 3: Restriction set: {(time = 14.30, relevance = 80)}

	Activity	Time	Semantic distance	Duration
Route 1	work	10:43:54	143.481	33 min
Route 2	leisure	16:57:32	146.397	10 min
Route 3	work	10:09:55	158.360	59 min
Route 4	leisure	17:01:36	159.620	36 min
Route 5	leisure	16:59:48	160.835	38 min

The value of the percentage of routes with a semantic distance lower than 160.835 was 0 for this restriction set although there were some other routes that matched it but with higher semantic distances.

Experiment 4: Restriction set: {(coordinates = (39.469794,-0.377655), relevance = 80)}

	Activity	Time	Semantic distance	Duration
Route 1	leisure	16:32:55	88.884	3h 29 min
Route 2	leisure	18:06:41	109.770	4h 57 min
Route 3	work	10:09:55	127.665	59 min
Route 4	work	8:05:58	236.014	1h 1 min
Route 5	leisure	22:20:07	523.421	22 min

The value of the percentage of routes with a semantic distance lower than 523.421 was 0 for this restriction set although there were some other routes that matched it but with higher semantic distances.

Experiment 5: Restriction set: {(time = 14:30, relevance = 80), (coordinates = (39.469794,-0.377655), relevance = 80)}

	Activity	Time	Semantic distance	Duration
Route 1	leisure	16:32:55	263.616	3h 29 min
Route 2	work	10:09:55	310.026	4h 57 min
Route 3	leisure	18:06:41	401.652	59 min
Route 4	work	8:05:58	517.638	1h 1 min
Route 5	leisure	22:20:07	908.696	22 min

The value of the percentage of routes with a semantic distance lower than 908.696 was 0 for this restriction set although there were some other routes that matched it but with higher semantic distances.

4.1 Experimental comparison

We have presented the results of the experiments carried out to confirm that the SACO behavior was correct. Next we will proceed to compare experimentally SACO with other algorithmic approaches for graph-based searching. To this end, we have selected two algorithms that have been vastly used in literature Simulated Annealing [27] and Tabu Search [28]. Both algorithms have been used to solve graph-based optimization problems [29] [30] with similar characteristics to the one presented here. These two alternative algorithmic approaches have been widely applied in several domains as it is discussed in [31].

Simulated Annealing (SA) is inspired by a natural process. SA emulates the physical process of slow cooling a mineral follows to reach a state of minimum energy, which corresponds with a perfect crystal, to perform the search related to the resolution of a minimization problem. One of the most important features of SA is that, during its operation, it can accept as solution of an iteration one that is worse than the best solution found so far. This behavior allows SA to avoid ending the search in a local optimum.

Tabu Search (TS) is an algorithm that allows solving combinatorial problems, as SA. It works by maintaining a current solution for the problem that is gradually improved by generating neighborhoods of solutions. TS assess every selected solution from the neighborhood and, if it is a better solution, it is added to a *tabu list* so it cannot be selected again. This eases to avoid the convergence to local optima.

In order to perform the experimental comparison of the algorithms a corpus of synthetically generated contextual sequences was built. The contextual sequences contain values for the terms of the ontology that it is been used in this paper. With this corpus, the behavior of the algorithms for different sizes of the problem search space (i.e. the user model) can be evaluated. Besides, some restriction sets have been defined, each algorithm was run 5 times for each restriction set and problem size, and both the average needed time and the average semantic distance of the best contextual sequence found were registered. With the registered average time, we can evaluate and compare the scalability of the algorithms. Regarding the average semantic distance, it is useful to assess whether the quality of the solutions found degrades as the size of the problem increases.

Restriction set #1

The first restriction set defined and evaluated was:

$$R1 = \{(activity = work, relevance = 80), (time = 8.00, relevance = 80)\}$$

As it may be observed in Figure 6, the average time to obtain solutions by the SACO algorithm is significantly lower and increases in a linear manner which demonstrates that our approach scales better than the two proposed alternative search algorithms. As an example, for model instances consisting of 2000 contextual sequences SACO is 82.19% faster than TS and 70.36% faster than SA.

Additionally, in Figure 7 it is plotted the average semantic distance (understood as a quality measure, the lower the distance, the better the solution quality) of the best contextual sequence found by the algorithms after each run. Results show that there are no significant differences among the average quality of the found solutions. Therefore, we may say that our approach, based on ant colony optimization is as effective as SA and TS in terms of finding good solutions to the problem but following a more effective strategy in terms of computational costs when the problem size is increased.

Restriction set #2

The second restriction set defined and evaluated was:

$$R2 = \{(coordinates = (39.469794,-0.377655), relevance = 80)\}$$

In this restriction set just one term has been included, the geographical coordinates. This means that the problem search space cannot be reduced as much as in the previous one because there are no relevant semantic terms, other than location, provided by the user in the restriction set. This would be the less favourable situation for our algorithm because there is no previous reduction of the search space based on semantic information. However, as it may be observed in Figure 8, our algorithm is still competitive and obtains solutions in an effective way with respect to SA and TS. The time required to obtain a solution by SACO grows linearly as the size of the problem is increased. In this particular example, for problem instances consisting of 2000 contextual sequences SACO is 89.28% faster than TS and 68.24% faster than SA and we observe a similar degradation, as in the previous example, of the required computation time when the problem size increases for TS and SA.

Figure 9 shows the average semantic distance of the best contextual sequences found by the algorithms after each run to find solutions for restriction set 2. As in the case of the first restriction set, there are no significant differences among the average quality of the solutions found by the algorithms. Although in this case some differences exist, they are very small compared with the total value of the semantic distance and should be attributed to the stochastic nature of this kind of algorithms and also to both the amount and type of the terms included in the restriction set.

5. CONCLUSIONS

The convergence of technological breakthroughs in cellphones and other portable devices allows us to keep in touch with others at all times and has also led to the birth of new services and applications, such as those based on the current user location. A natural and convenient way of providing users with information on the nearby services available can be achieved by means of map navigation systems. However, at the present time users have to manually filter out non-interesting services in order to reduce the amount of information displayed on the map. In this paper we have presented a semantic multicriteria ACO-based algorithm designed to deal with this problem. Our algorithm learns the user's routes, adds the associated context information and then predicts the most likely route the user is following, given his current location and context data. This knowledge could be used to present the user with the nearby available services related to his current (or most likely future) context data. In this way users are not forced to manually filter out services and still obtain a non-bloated map visualization. Besides, SACO is not forced to use a prefixed set of context variables as in other existing works, what allows users to define their own ontologies with individual meaningful context variables that the algorithm will use. This fact also allows users to easily and meaningfully ask for a set of likely routes for several given context data values. We have carried out a comparison in this paper between SACO and two alternative algorithms widely used for graph-based searching: Tabu-Search and Simulated Annealing. The results showed that SACO obtains high quality solutions whose semantic distance to the restriction set is on average as low as those obtained by using SA and TS but with a significantly lower computation cost. It has been also observed that the computational cost of our approach increases linearly as the size of the problem increases which makes it appropriate when dealing with large size problem instances.

We also developed an application to interact with the algorithm and display the returned routes and have performed several experiments that show that multiple terms of the ontology can be used to obtain a set of solutions. The results also show that the algorithm behaved as expected and did not omit suitable routes from the solution space.

Our plans for future work include increasing the expressivity of restriction sets by using temporal logic operators, such as *after*, *until* or *next* between instances of a certain ontology term, also performing consumer tests to assess the usefulness and acceptance of a map navigation system based on the proposed algorithm, and exploring the use of hybrid metaheuristics in order to assess whether results may improve (either the quality of the solutions or the computation time). In particular, our algorithm allows the integration of optional local search heuristics, as stated in section 3.1. Previous works on hybrid metaheuristics such as [15] [32] [33] have shown that this kind of combination may lead to improvements in the obtained results. However, in our case additional experiments would be needed to evaluate

the improvement obtained and its trade-off with respect to the required additional computational work because, as we concluded in [16], the nature of the problem or the design of the approach may reduce the room for improvement while incurring in significant computational costs.

6. ACKNOWLEDGMENTS

This work has been supported by the Spanish Ministry of Science and Innovation, Centre for the Development of Industrial Technology (CDTI) under the funding project CENIT-MIO! CENIT-2008 1019.

7. REFERENCES

- [1] Y. Zheng, L. Liu, L. Wang, X. Xie, Learning Transportation Mode from Raw GPS Data for Geographic Applications on the Web, In: Proc. of the 17th international conference on World Wide Web 2008, pp. 247-256.
- [2] K. Virrantaus, J. Markkula, A. Garmash, Y.V. Terziyan, Developing GIS-Supported Location-Based Services. In: Proc. of WGIS'2001 – First International Workshop on Web Geographical Information Systems., Kyoto, Japan, 2001, pp. 423–432.
- [3] A. Picón, S. Rodríguez-Vaamonde, J. Jaén, Jose A. Mocholi, D. García, A. Cadenas, A statistical recommendation model of mobile services based on contextual evidences, *Expert Systems with Applications*, 39(1), 2012, pp. 647-653.
- [4] TomTom IQ Routes. <http://www.tomtom.com/page/iq-routes>, last visited on July 29, 2010.
- [5] R. Hariharan, K. Toyama, Project Lachesis: Parsing and modeling location histories, In: R. Hariharan, K. Toyama (Eds.), Proc. of GIScienc'04, Springer Press, 2004.
- [6] R. Simmons, B. Browning, Z. Yilu, V. Sadekar, Learning to Predict Driver Route and Destination Intent, In: Proc. of Intelligent Transportation Systems Conference, IEEE, 2006, pp.127-132.
- [7] D. Ashbrook, T. Starner, Using GPS to learn significant locations and predict movement across multiple users, *Personal and Ubiquitous Computing* 7(5), 2003, pp. 275-286.
- [8] J. Krumm, E. Horvitz, Predestination: Inferring Destinations from Partial Trajectories, In: Proc. of UBICOMP'06, Springer-Verlag Press, California, 2006, pp. 243-260.
- [9] N. Marmasse, C. Schmandt, A User-Centered Location Model, *Personal and Ubiquitous Computing* 6.5-6, 2002.
- [10] J. Froehlich, J. Krumm, Route Prediction from Trip Observations, In: Society of Automotive Engineers (SAE) World Congress, Paper 2008-01-0195, Detroit, Michigan USA, 2008.
- [11] N. Persad-Maharaj, S.J. Barbeau, M.A. Labrador, P.L. Winters, R. Perez, N.L. Georggi, Real-time Travel Path Prediction using GPS-enabled Mobile Phones, In Proc. of the 15th World Congress on Intelligent Transportation Systems, New York, 2008.

- [12] L. Liao, D.J. Patterson, D. Fox, H. Kautz, Building Personal Maps from GPS Data, In: Proceedings of IJCAI Workshop on Modeling Others from Observation, 2005.
- [13] K. Torkkola, K. Zhang, H. Li, H. Zhang, C. Schreiner, M. Gardner, Traffic Advisories Based on Route Prediction, In: Workshop on Mobile Interaction with the Real World, Singapore, 2007.
- [14] A. Catala, J. Jaén, J.A. Mocholí, Strategies for Accelerating Ant Colony Optimization Algorithms on Graphical Processing Units, In: Proc. of the 2007 IEEE Congress on Evolutionary Computation, 2007, pp. 492-500.
- [15] J. A. Mocholí, J. Jaén, J.H. Canós, A Grid Ant Colony Algorithm for the Orienteering Problem. In: Proc. of the 2005 IEEE Congress on Evolutionary Computation, Vol. 1, 2005, pp. 942-949.
- [16] J. Jaén, J. A. Mocholí, A. Catalá, E. Navarro, Digital ants as the best cicerones for museum visitors, Appl. Soft Comput. 11(1), 2011, pp. 111-119.
- [17] S. Goss, S. Aron, J.L. Deneubourg, J.-M. Pasteels, Self-organized shortcuts in the Argentine ant, *Naturwissenschaften*, 76, 1989, pp. 579-581.
- [18] M. Dorigo, V. Maniezzo, A. Colomi, The Ant System: Optimization by a Colony of Cooperating Agents, *IEEE Trans. Systems, Man and Cybernetics, Part B*, 26, 29-4, 1996.
- [19] M. Dorigo, T. Stützle, The ant colony optimization metaheuristic: Algorithms, applications and advances, In: F. Glover, G. Kochenberger (Eds.), *Handbook of Metaheuristics*, Kluwer Academic Publishers, 2003, pp. 251-285.
- [20] Y.C. Liang, A.E. Smith, An Ant Colony Approach to the Orienteering Problem, *Journal of the Chinese Institute of Industrial Engineers*, Vol. 23, no. 5, 2003, pp. 403-414.
- [21] M. Dorigo and L. M. Gambardella, Ant colony system: A cooperative learning approach to the traveling salesman problem, *IEEE Transactions on Evolutionary Computation*, 1(1), 1997, pp. 53-66.
- [22] A.F. Smeaton, A. Quigley, Experiments on using semantic distances between words in image caption retrieval, In: Proc. of the International ACM SIGIR Conference on Research and Development in Information Retrieval, 1995.
- [23] J. Gonzalo, F. Verdejo, I. Chugur, J. Cigarran, Indexing with wordnet synsets can improve text retrieval, *Coling-ACL'98 Workshop: Usage of WordNet in Natural Language Processing Systems*, 1998, pp. 38-44.
- [24] L. Khan, D. McLeod, Audio structuring and personalized retrieval using ontologies, In: *IEEE Advances in Digital Libraries*, 2000.
- [25] M.A. Bunge, *Treatise on basic philosophy: ontology: the furniture of the world*, Reidel, Boston, 1977.
- [26] Q. Duan, T. W. Liao, Improved ant colony optimization algorithms for determining project critical paths, *Automation in Construction*, 19 (6), pp. 676-693.
- [27] S. Kirkpatrick, C. D. Gelatt, M. P. Vecchi, Optimization by Simulated Annealing, *Science* 220 (4598), 1983, pp. 671-680.

- [28] F. Glover, Tabu Search, Part I, *ORSA J. Computing*, 1, 1989, pp. 190-206.
- [29] Xiutang Geng, Zihua Chen, Wei Yang, Deqian Shi, Kai Zhao, Solving the traveling salesman problem based on an adaptive simulated annealing algorithm with greedy search, *Appl. Soft Comput.* 11(4), pp 3680-3689.
- [30] I. H. Osman, Metastrategy simulated annealing and tabu search algorithms for the vehicle routing problem, *Annals of Operations Research*, 41, 1993, pp. 421–451.
- [31] I. H. Osman, G. Laporte, Metaheuristics: A bibliography, *Annals of Operations Research*, 63, 1996, pp. 513-623.
- [32] T. W. Liao, Two hybrid differential evolution algorithms for engineering design optimization, *Appl. Soft Comput.* 10 (4), pp. 1188-1199.
- [33] T. W. Liao, P. J. Egbelu, P. C. Chang, Simultaneous dock assignment and sequencing of inbound trucks under a fixed outbound truck schedule in multi-door crossdocking operations, *International Journal of Production Economics*, in press, 2012.

List of Figures

Figure 1. The SACO algorithm process illustrated.

Figure 2. Conceptual model of SACO.

Figure 3. The process of adding a contextual sequence to a user model.

Figure 4. Entering values and their relevance factors to create a restriction set.

Figure 5. Visualization of routes returned for a certain restriction set.

Figure 6. Average time necessary to obtain solutions for the restriction set R1 from each algorithm and for each problem size.

Figure 7. Average semantic distance of the best solution found by each algorithm for the restriction set R1 and for each problem size.

Figure 8. Average time necessary to obtain solutions for the restriction set R2 from each algorithm and for each problem size.

Figure 9. Average semantic distance of the best solution found by each algorithm for the restriction set R2 and for each problem size

List of Tables

Table 1. Characteristics comparison of related works and SACO.

Figure 1
[Click here to download high resolution image](#)

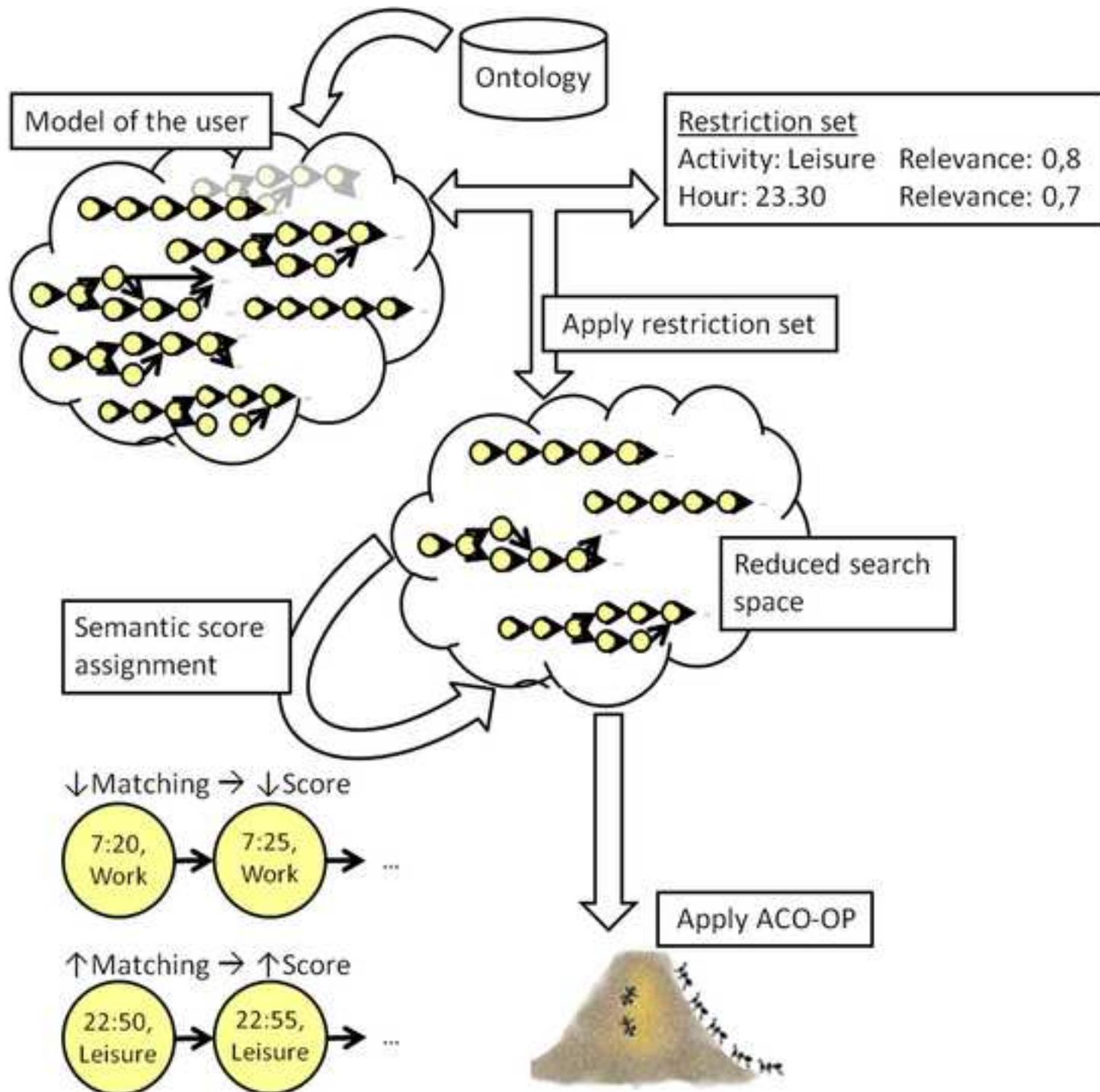


Figure 1 BW

[Click here to download high resolution image](#)

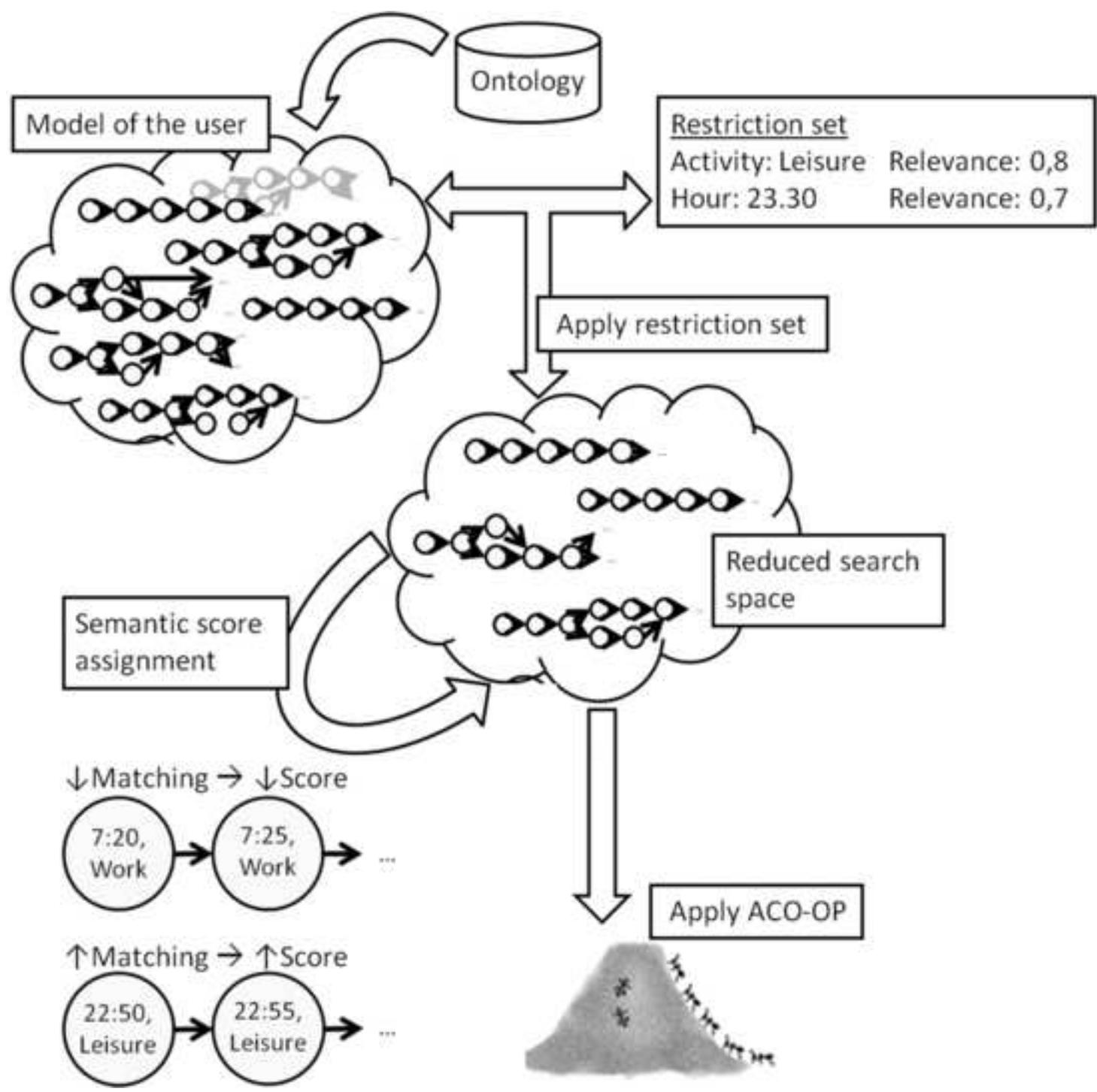


Figure 2
[Click here to download high resolution image](#)

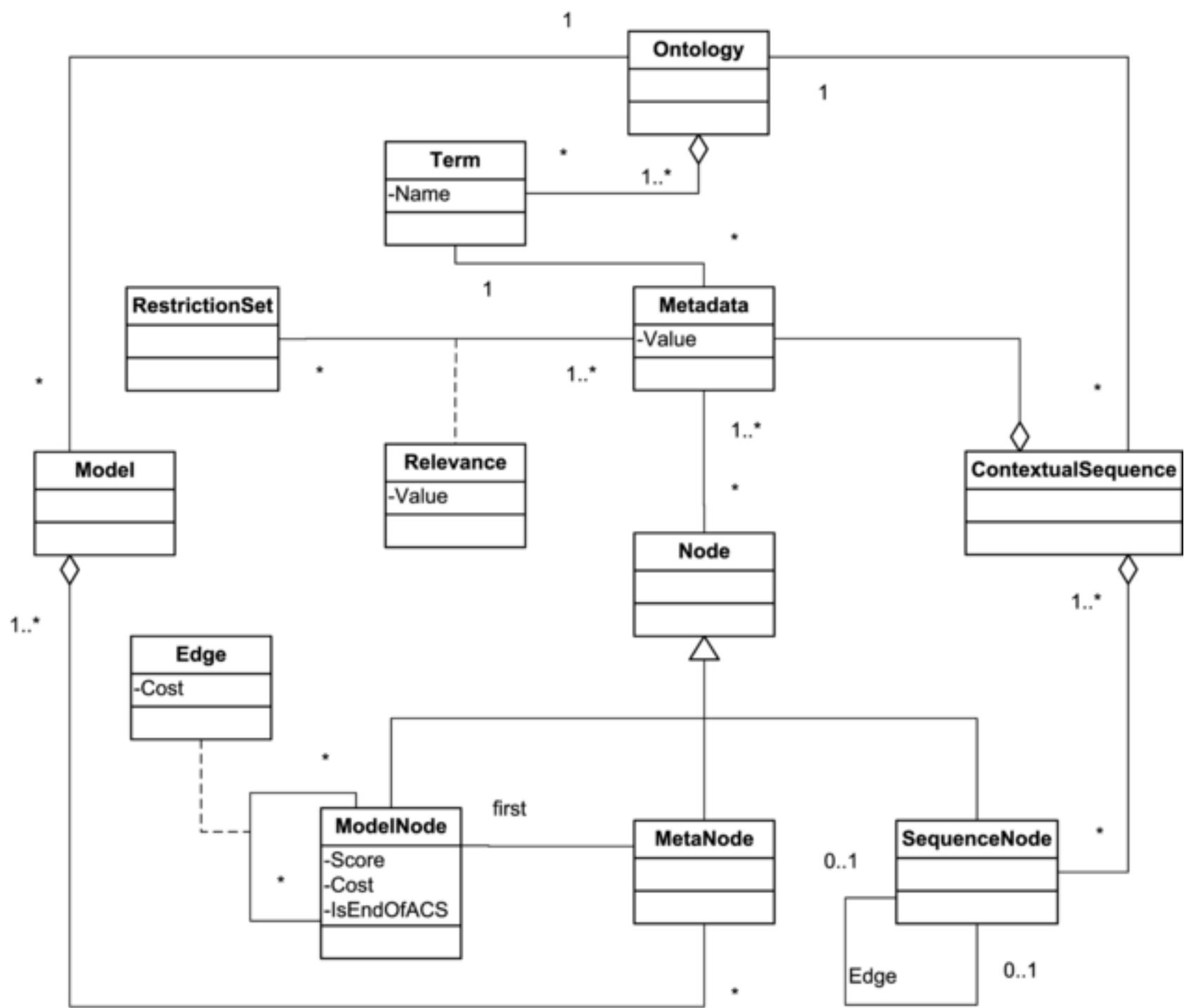


Figure 3

[Click here to download high resolution image](#)

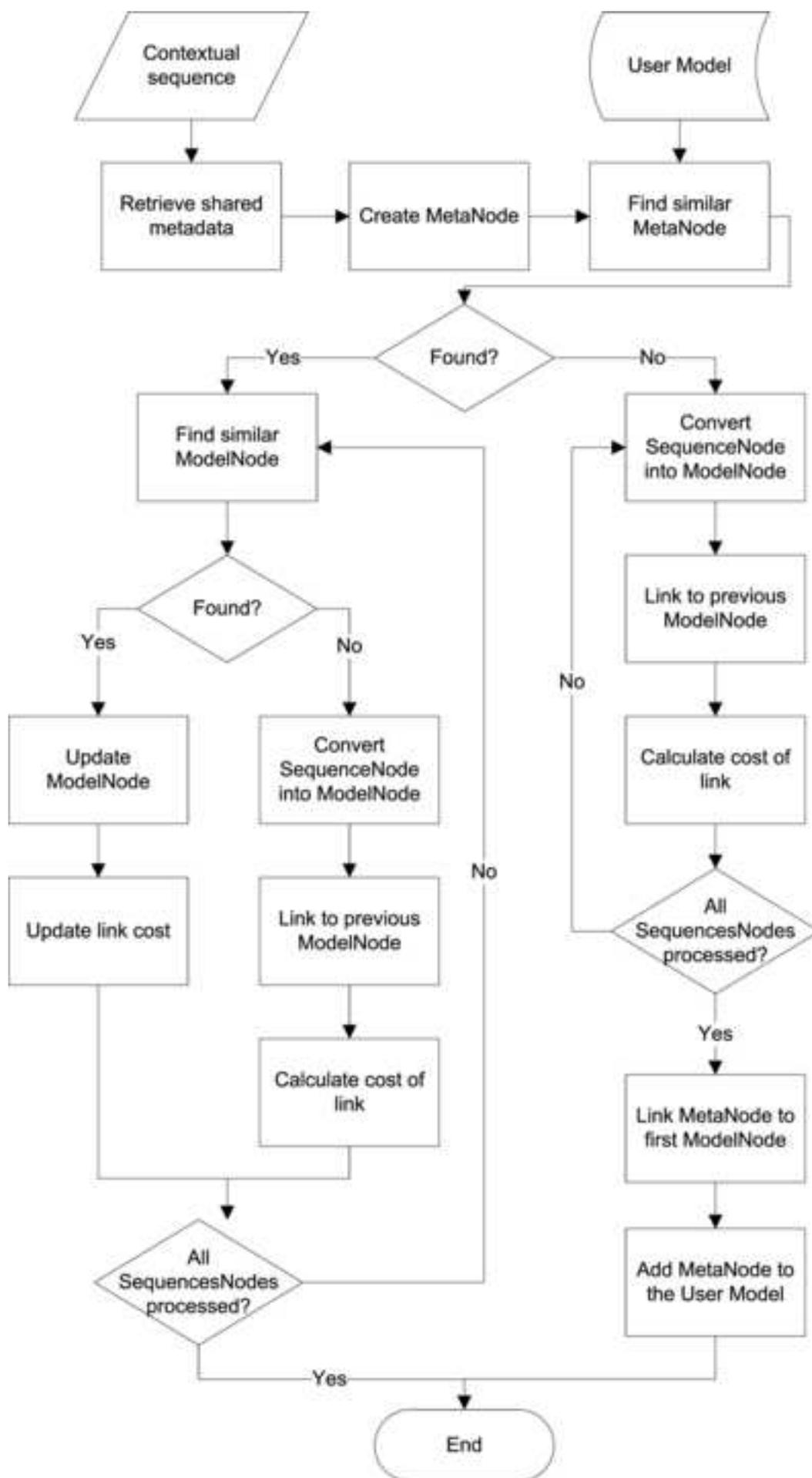


Figure 4
[Click here to download high resolution image](#)

Seleccionar usuario Nueva Consulta | Mostrar Capas ▾ | Limpiar Mapa

Metadato Valor Relevancia

Num. soluciones Num. marcas

Metadatos	Valor	Relevancia	<input type="button" value="Eliminar Elemento"/>
Actividad	Trabajo	80	
Hora	8:00	80	



Figure 4 BW

[Click here to download high resolution image](#)

Seleccionar usuario Nueva Consulta | Mostrar Capas ▾ | Limpiar Mapa

Metadato Num. soluciones

Valor Num. marcas

Relevancia

Metadatos	Valor	Relevancia	<input type="button" value="Eliminar Elemento"/>
Actividad	Trabajo	80	
Hora	8:00	80	



Figure 5
[Click here to download high resolution image](#)

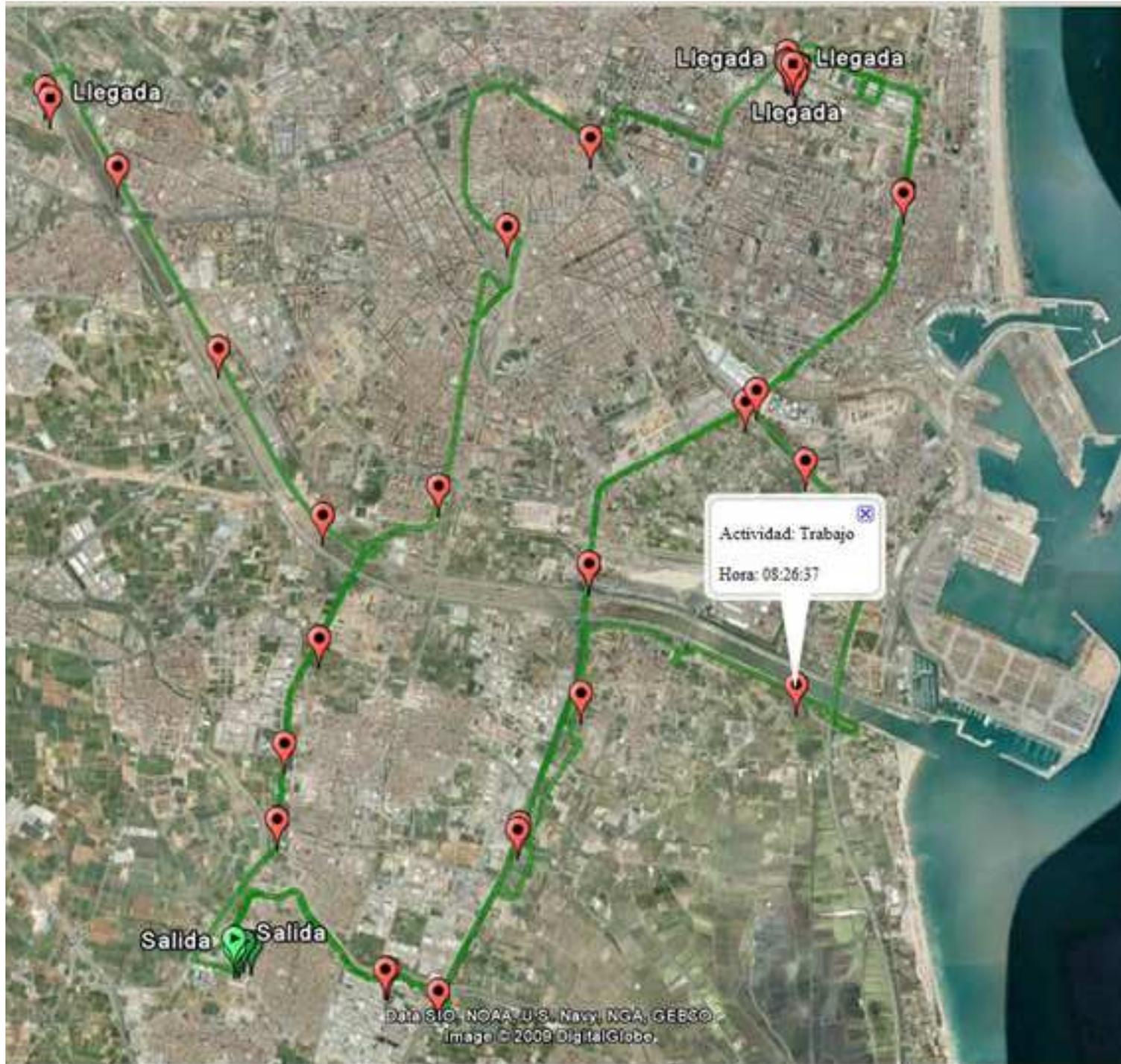


Figure 5 BW

[Click here to download high resolution image](#)



Figure 6

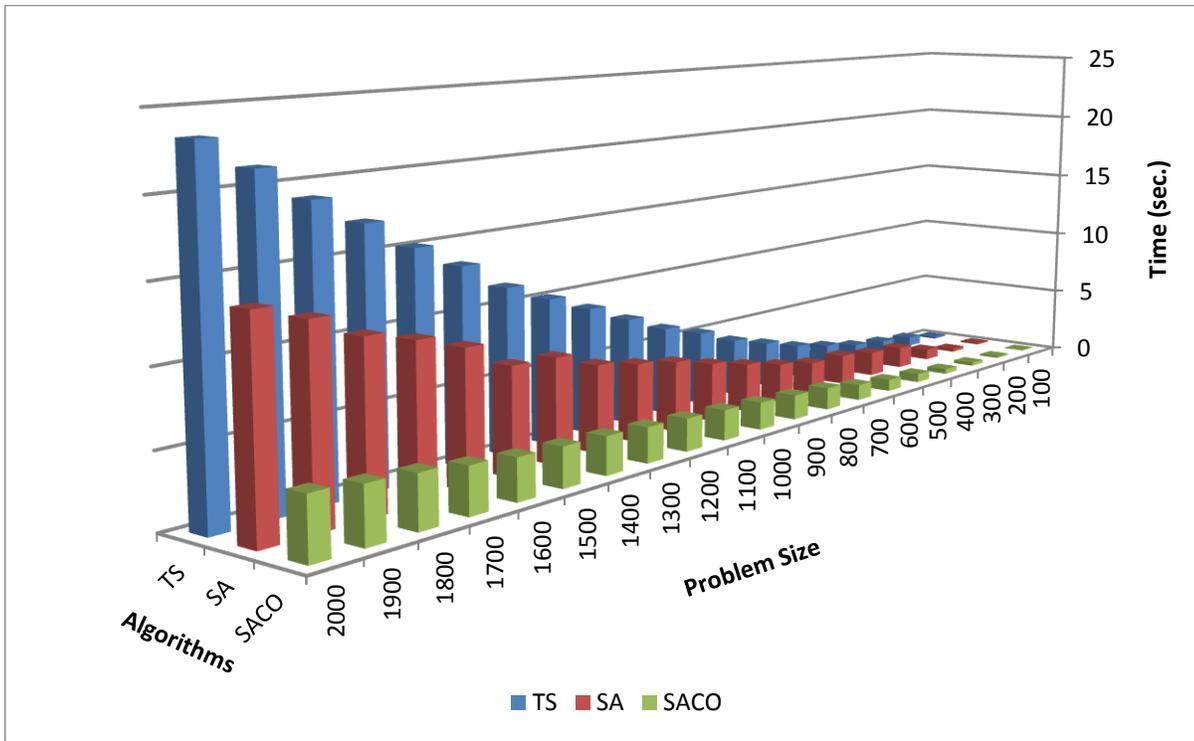


Figure 7

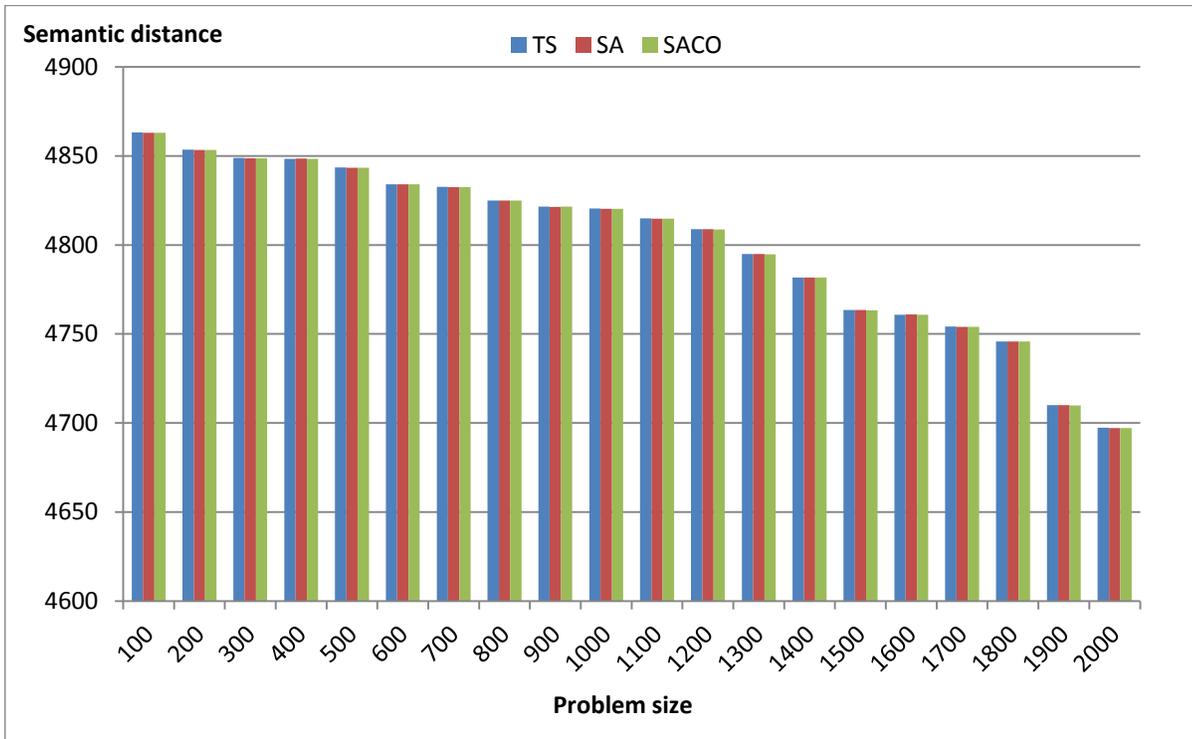


Figure 8

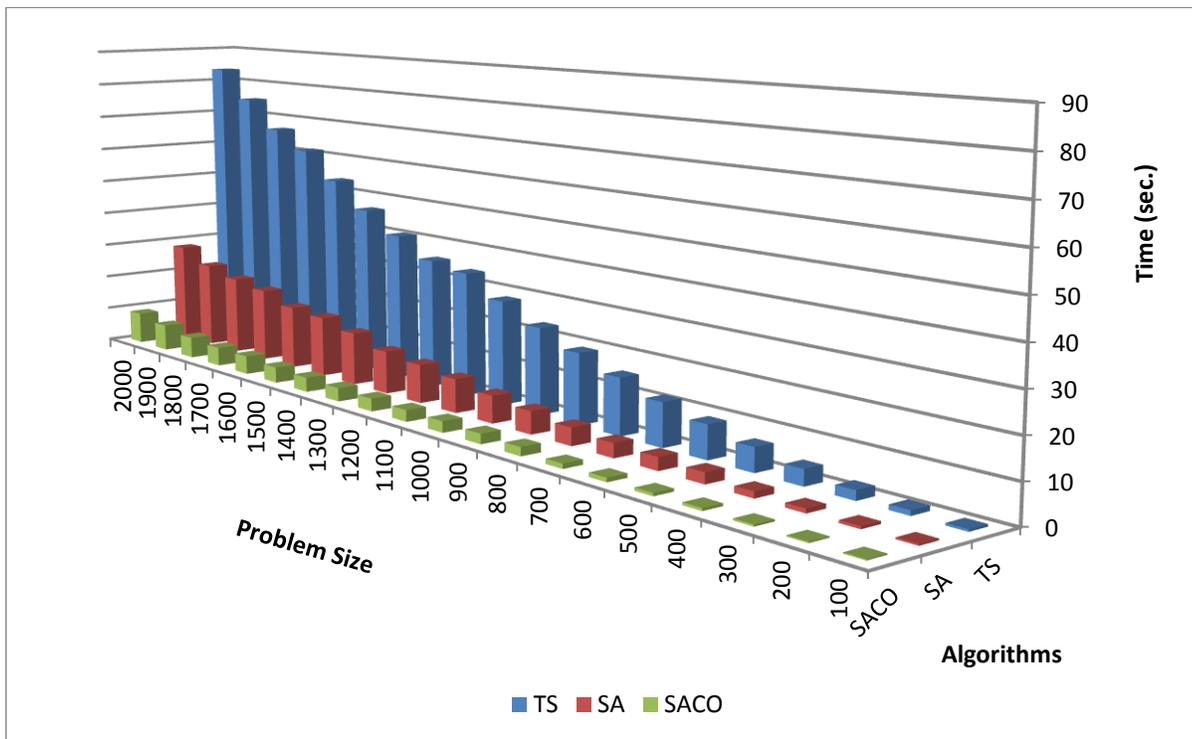


Figure 9

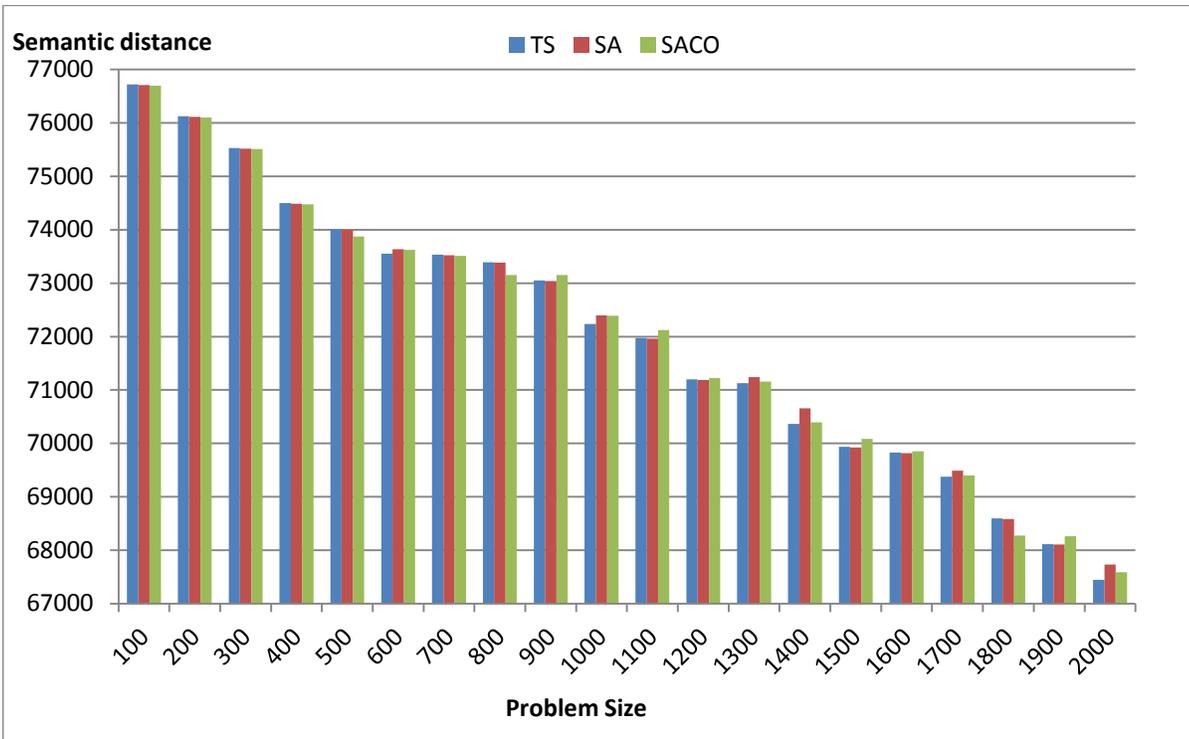


Table 1

Project	System predicts ...	Context data used	Continuous learning	Semantic searching	Prediction technique
Lachesis [5]	Destination	GPS	--	No	Markov model
Simmons et al. [6]	Route	GPS	--	No	Markov model
Ashbrook et al. [7]	Destination	GPS	--	No	Markov model
Predestination [8]	Destination	GPS	Yes	No	Bayes formula
comMotion [9]	Destination	GPS	Yes	No	Bayes classifier, Markov model, and histogram matching
Froehlich et al. [10]	Route	GPS	--	No	Similarity distance ad-hoc algorithms
Persad-Maharaj et al. [11]	Route	GPS	Yes	No	Spatial database
Liao et al. [12]	Destination	GPS, User activity	--	No	Hierarchical dynamic Bayesian network
Torkkola et al. [13]	Routes	GPS, GSM cell ID, mobile phone related info.	Yes	No	Similarity distance ad-hoc algorithm
SACO	Routes	Any domain-defined ontology	Yes	Yes	Ant Colony Optimization