TOURISM-KM, A variant of MMKP applied to the tourism domain

Romain Picot-Clémente, Florence Mendes, Christophe Cruz, Christophe Nicolle

TOURISM-KM

A variant of MMKP applied to the tourism domain

Romain Picot-Clémente, Florence Mendes, Christophe Cruz and Christophe Nicolle

Laboratoire Le2i, UFR Sciences et Techniques Universit de Bourgogne, B.P. 47870, 21078 Dijon, France

{*romain.picot-clemente, florence.mendes, christophe.cruz, christophe.nicolle*}

@u-bourgogne.fr

Keywords: Knowledge Management ; Simulated Annealing ; Knapsack Problem ; Decision Support

Abstract: We are interested in an original real-world problem coming from tourism field. We describe a modelling of the problem and propose a first approach that mixes knowledge management and operational research methods. Our algorithms have been implemented in order to produce tourism solutions that are not unique for a given request but that take into account the preferences of the tourist user and provide a personalized solution. We report computational results obtained on real-world instances.

1 INTRODUCTION

The tourism area has widely changed over last years, thanks to innovative services accessible via internet and smartphones. Tourism offer is so abundant that the customer may feel lost if the information is not clearly ordered. We present a method to help the tourist to plan his stay, which uses knowledge management techniques to take into account his preferences and his proper specifications (in couple or with children, dog, etc.). Our goal is to suggest an offer that matches at best the expectations of the customer. In this purpose, we take into account at the same time adequacy between tourism offer and customer specifications, interest of the customer for this offer, and also geographical distance between the different elements of the offer.

In this paper, we present the resolution of a hard holiday scheduling problem by using a combination of metaheuristics based on simulated annealing with a semantic modelling of tourism knowledge focused on users. This work takes place under the (CHECKSEM,) project with the aim of providing a comprehensive set of knowledge management methods and tools focused on the user. Some of these methods have already been applied successfully in several domains such as civil engineering (Cruz and Nicolle, 2006) or archaeology (Karmacharya et al., 2009).

Our paper is organized as follows. In section 2, we briefly present the concepts related to the domain models, the definition of a valid solution to the problem of best combination, as well as how to measure the quality of a solution. In part 3, we present a stochastic algorithm based on simulated annealing. Finally, we discuss the effectiveness of our approach and opportunities for further developments.

2 PROBLEM DEFINITION

The proposed system consists of four layers: a model of goals, a domain model, a user model and an adaptation model. The user model represents the user in the system, it consists of the goals of the user for his future stay. The domain model is a domain ontology in which all the tourism knowledge useful for the application program is defined. The goals model contains all possible goals of users, they are linked with the domain model by first order rules. This layered modelling is detailed in (Picot-Clemente et al., 2010).

In the following paragraphs, we define only the concepts that are necessary to introduce our optimization problem.

2.1 Items and Weights

A tourist is represented in our system by a *user profile*, composed of goals from the goals model and defining the objectives of the associated user. User profiles are stocked into a *users model*. In the following, we will consider only one user, supposing that the process to produce a solution can be run independently for each user of the system.

Let Ty bet the set of possible *activity types* in the system. For example, restaurant, museum, or hotel, are standard activity types.

Let I be the set of all the elements that can be supplied by the system in the final recommendation. An *item* $i \in I$ is defined by a vector composed of a name $i.name$, a type $i.type \in Ty$, and geographical coordinates $i.x$ and $i.y$.

$$i = i.name, i.type, i.x, i.y \quad (1)$$

The *domain model* represents the domain knowledge, using an ontology composed of concepts, relations between concepts, and individuals. Valid items are integrated into the ontology as individuals. For convenience in this paper we reduce the domain model to the set I of valid items :

$$I = i_1, ..., i_{nbi} \quad (2)$$

The *weight* of an item $i \in I$, denoted w_i, represents the interest of this item for the user. The method used to compute the weights of items consists of a propagation of weights into the ontological domain model, by following the different goals that are defined by the user (see (Picot-Clemente et al., 2010) for further details).

2.2 Patterns and Combinations

When asking for a holiday proposal, the user chooses a solution *pattern*, which defines the types of items authorized in a solution.

$$P = (ty_1, ty_2, , ty_{NbTy} / \forall 1 \leq j \leq NbTy, t_j \in Ty) \quad (3)$$

A valid *combination* C_k for a pattern P is a set of items from the domain model, chosen under the constraints given by the combination pattern.

$$C_k = (i_1, i_2, , i_N), \quad \forall 1 <= j <= Ni_j.type = ty_j, ty_j \in P \quad 1 \leq k \leq NbC \quad (4)$$

All valid combinations for P can be ordered from C_1 to C_{NbC}. The *weight of a combination* C_k, denoted W_{C_k}, equals the average weight of each item of the combination. It represents the global interest of the user for the combination, without considering distance between items.

$$W_{C_k} = \frac{\sum_{j=0}^{N} w_{i_j}}{N} \quad (5)$$

In addition with the main combination pattern, the user as the possibility to add more requirements via the definition of *subcombinations patterns*. Example : Let $p_1 = Museum, Hotel$ and $p_2 = Restaurant, Hotel$ be two subcombination patterns, corresponding to the main pattern $P = Museum, Restaurant, Hotel$. The combination $C_3 = Louvremuseum, RestaurantduPalais, HotelJacques$ can generate two subcombinations corresponding to the two subpatterns : $C_{3,1} = Louvremuseum, HotelJacques$ and $C_{3,2} = RestaurantduPalais, HotelJacques$

2.3 Dispersion and Relevance of a Solution

The problem of finding the best combination of items for a given user requires to introduce some geographic properties to distinguish and compare the solutions. Indeed, weight of an item is not sufficient to evaluate the quality of a combination. In this paragraph we define a method to evaluate the combination relevance by taking into account the distance between items.

For a given combination $C_{j,u}$, the *dispersion* $\sigma(C_{j,u})$ equals the standard deviation of the coordinates of combination items.

$$\sigma(C_{j,u}) = \sqrt{\frac{\sum_{i=0}^{|C|-1}((i.x - \frac{\sum_{i=0}^{|C|-1}(i.x)^2}{|C|}) + (i.y - \frac{\sum_{i=0}^{|C|-1}(i.y)^2)}{|C|}))^2}{|C|}} \quad (6)$$

So for each combination, the dispersion allows us to quantify the geographical distance between combination items. A same dispersion can be considered differently by two users, so we define a dispersion tolerance, depending on the concerned user and the combination pattern. The *dispersion tolerance*, denoted $Tol(P)$ is a number representing the tolerance of the user in terms of geographical distance between items for pattern P.

The *moderated dispersion* is used to take into account the same dispersion value for combinations that are in the same order of distance, according to the dispersion tolerance of the user :

$$\sigma_{mod}(C_k) = \lceil \frac{\sigma(C_k)}{Tol(P)} \rceil \quad (7)$$

By this way, two combinations with dispersion values varying only by a few meters will not be distinguished.

As for combinations, each subcombination is associated with a dispersion value and a moderated dispersion value. The *relevance* of a combination C_k is an aggregate of W_{C_k} and moderate dispersion of C_k and its subcombinations :

$$\Re(C_k) = \frac{W_{C_k}}{\sigma_{mod}(C_k) + \sum_{l=0}^{S-1} \sigma_{mod}(C_{k,l})} \quad (8)$$

where S is the number of subpatterns associated to the pattern P.

Definition 1. *Tourist Problem : Given a user, a set of items and patterns to compose a valid combination of items, the problem of finding the best items combination consists in finding for each user a combination with maximal relevance :*

$$maximise\ \Re(C_k), \quad with\ 1 \leq k \leq NbC, \quad (9)$$

2.4 Links With Other Optimization Problems

This problem can be viewed as a Set Packing problem, as in (Avella et al., 2006). The authors propose a simplified variant of tourist problem, they call the Intelligent Tourist Problem, solved with a LP-based heuristic. Their modelling include items and weights for the items depending on user preferences, but they only take into account one type of item (tourism activities). So they don't follow any pattern of combination, but add items to a combination if the time period necessary for the activity matches a free time period for the tourist.

Our problem has similarities with a combinatorial optimization problem called the Knapsack Problem (Karp., 1972). The knapsack problem derives its name from the problem faced by someone who is constrained by a fixed-size knapsack and must fill it with the most useful items. Each item as a weight and a value. Items put in the knapsack have to maximise the total value without exceeding a given maximum weight. A common formulation of this problem, with each item different from the others is as follows (0-1 Knapsack Problem):

$$maximise \sum_{i=1}^{n} v_i x_i \quad (10)$$

subject to

$$\sum_{i=1}^{n} w_i x_i \leq W, x_i \in 0,1 \quad (11)$$

where v_i and w_i are respectively the value and the weight of item i. W is the maximum weight of the knapsack. x_i equals 1 if item i is put into the knapsack, 0 otherwise.

This problem has been proven to be NP-complete (Lagoudakis, 1996). Several variants of the knapsack problem have been studied : the multidimensional knapsack, the quadratic knapsack problem, etc. (see (Martello and Toth, 1990) for a survey). Our tourism problem can be viewed as a variant from the multi-choice multi-dimensional knapsack problem (MMKP). In standard MMKP, items are grouped in function of their type and only one representant

of each type has to be chosen. In our problem several items of the same type can be chosen, depending on the combination and subcombination patterns. MMKP is NP-hard, so it would not be efficient to apply an exact method to solve it, especially for a real-time decision making application (Chen et al., 1999). Recent heuristic approaches use various techniques, such as Local Search (Hifi et al., 2004; Hifi et al., 2006), Tabu Search and Ant Colony Optimization (Lau and Lim, 2004), or reductions of the search space (Akbar et al., 2006), etc.

In the next part, we propose a simulated annealing algorithm to find a good solution in reasonable time.

3 SIMULATED ANNEALING ALGORITHM

In order to solve the problem of finding the best combination of items from the domain model for a given user, we suggest the use of a stochastic algorithm called simulated annealing (Kirkpatrick et al., 1983) that is inspired from a method used in the steel industry.

3.1 Algorithm Description

The simulated annealing algorithm used here is based on this principle. At the beginning, the algorithm chooses an initial random combination of individuals following a given combination pattern (for instance, a combination consisting of a hotel, two restaurants and two activities). This combination has an energy $E0$, called the initial energy, which represents the quality of a combination. This energy of a combination is based on the relevance of the combination:

$$\varepsilon(C_k) = \frac{1}{\Re(C_k)} \tag{12}$$

The lower the energy is, the better the combination is. A variable T, called temperature, decreases in increments over time. At each level of temperature a certain number of elementary random changes is tested on the current combination. A cost df is associated to each modification; it is defined as the difference between the combinations energy after the modification and the one before. A negative cost signifies the current combination has a lesser energy than the previous one (thus better by definition), it is then kept. Conversely, a positive cost represents a bad change. Nevertheless, it can be kept according a given probability (acceptance rate) depending on the current level of temperature and the cost. The higher the temperature is, the higher the probability is. Thus, over time, the number of changes allowed decreases as the temperature decreases, until no longer accepting any changes. Finally, the system is said frozen, and the current combination becomes the final combination to be presented to the user. The acceptance rate is defined in (13) where Tk is the temperature at the level $k, k \in N$.

$$T_a = e^{-\frac{d_f}{T_k}} \tag{13}$$

The temperature decrease is achieved through a geometric decay at each level:

$$T_k = g(T_{k-1}) = coef * T_{k-1} = coef^k * T_0 \tag{14}$$

where k is the current level and $0 < coef < 1$.

3.2 Algorithm Parameters

The arbitrary definition of the various parameters of simulated annealing algorithm is the main disadvantage of this algorithm. First, we have to establish the initial temperature T_0. If T_0 is too high, the first modifications will all be accepted without considering the quality of the solutions, which is a waste of time. Inversely, if T_0 value is too small, the exploration of the solutions space will be too limited. We have to find a medium value. A way to find this value consists in generating some expensive alterations and computing the medium variation $d_f moy$. A first acceptation rate is chosen (we use (0.9) in our experiments), the value of T_0 is computed :

$$T_0 = \frac{d_{fmean}}{ln\frac{1}{T_a}} \tag{15}$$

Then we have to choose the temperature decreasing coefficient and the number of iterations at each step. The decreasing coefficient is included between 0 and 1. The higher is the coefficient, the slower is the

Algorithm 1 Simulated annealing algorithm

Input: I items and their weights for the current user, combination patterns

Output: a valid combination with minimal Energy

C_0 = initial valid solution with energy $\varepsilon 0$
$T = T_0$
$k = 0$
while the system is not frozen **do**
  $k++$
  Obtain C_k by an elementary transformation of C_{k-1}
  Compute energy ε_k
  $df = \varepsilon_k - \varepsilon_{k-1}$
  if $df > 0$ **then**
    use an acceptance rate to randomly accept or refuse the modified solution as current solution ;
  else
    The modified solution is accepted as current solution ;
  end if
  the temperature is lowered ;
end while

decreasing. A value of 0.6 has been chosen, allowing us to obtain good results in reasonable time. The number of iterations at each step determines the number of changes allowed for each temperature step. When this number is reached, we say that the system is in a statistical equilibrium and we start the temperature decreasing to a new step. In our experiments, this number has been fixed at 2000. Finally, the stop criteria is important. We say that the system is frozen when no more change is acceptable. In pratice, some researchers choose to stop the algorithm when the system reaches a fixed temperature or when a maximal number of steps has been exceeded. We choose to stop the execution when no change have been done during a fixed number of modifications (2000).

3.3 Experimental Results

In order to justify the use of this simulated annealing algorithm, instead of a simpler algorithm like a Hill-Climbing, some benchmarks have been realized on real and random datasets, using the simulated algorithm described in the previous part and a Hill-Climbing algorithm. The Hill-Climbing algorithm used is presented in algorithm 2

Algorithm 2 Hill-Climbing algorithm

Input: I items and their weights for the current user, combination patterns

Output: a valid combination with minimal Energy

R = initial valid solution randomly chosen
repeat
  Obtain S by an elementary transformation of R
  if $\varepsilon_S < \varepsilon_R$ **then**
    R=S
  end if
until X transformations without a change or the time limit is reached
return S

The real dataset includes 3724 items which are composed of activities, 1008 restaurants and 727 accommodations. The random dataset includes 30000 items composed of 10000 activities, 10000 restaurants and 10000 accommodations. The items coordinates are defined randomly into the bounding rectangle of a french department. The items weights are also defined randomly between 0 and 1000.

The table 1 compares the results obtained by the Simulated Annealing algorithm (SA) with those obtained by the Hill-Climbing algorithm (HC). It shows the average values and the average times got on the real and the random datasets. These averages are performed on 100 iterations of the algorithms.

Table 1: Simulated annealing VS Hill-climbing

	Random Dataset		Real Dataset	
	Energy $*10^6$	Time (ms)	Energy $*10^6$	Time (ms)
SA	199.56	239	169.05	285
HC	812.26	8	310.31	9

For the tourist real application, we have a constraint consisting in needing a generation of the combinations in near real-time. This time constraint corresponds to an arbitrary tolerance of 500 ms. Thus, given that the computation times are lower than this

threshold, the two methods are adequate. However, the comparative table shows the average energy of the combinations given by the simulated annealing is better (lower) than the one given by the Hill-Climbing. This difference of relevance is reflected in the propositions to the user. That is why the use of the simulated annealing algorithm is justified for the near real-time application we want.

4 CONCLUSIONS AND FUTURE WORK

This paper presented the resolution of a hard holiday scheduling problem by using a combination of metaheuristics based on simulated annealing with a semantic modelling of tourism knowledge focused on users. In order to solve the problem of finding the best combination of items from the domain model for a given user, we used a simulated annealing algorithm. This work is a part of a more generic project which aims to build a touristic recommender system. This work combines the Semantic Web technologies (mainly ontologies), the model of the adaptive hypermedia systems, and combinatory algorithms in order to provide recommendations. A recommendation is a combination of items formed according to a semantic pattern defined with the help of a domain ontology. This research project was developed in cooperation with a French tourism company called Côte d'or Tourisme. Since June 2011, a smartphone application was free of charge and available to users on the Apple store or android market. Now, we work to improve our result by using a multi-objective approach. One of the main difficulties of this improvement will be the obtainment of workable results despite a very short execution time allowed for the smartphone application.

REFERENCES

Akbar, M., Rahman, M., Kaykobad, M., Manning, E., and Shoja, G. (2006). Solving the multidimensional multiple-choice knapsack problem by constructing convex hulls. *Computers and Operations Research*, 33(5):12591273.

Avella, P., Dauria, B., and Salerno, S. (2006). A lp-based heuristic for a time-constrained routing problem. *European Journal of Operational Research*.

CHECKSEM. http://www.checksem.fr.

Chen, L., Khan, S., Li, K., and Manning, E. (1999). Building an adaptive multimedia system using the utility model. *Lecture Notes in Computer Science*, 1586:289–298.

Cruz, C. and Nicolle, C. (2006). Active3d : Vector of collaboration, between sharing and data exchange. *INFOCOMP, Jounal of Computer Science*, 5(3):1–8.

Hifi, M., Michrafy, M., and Sbihi, A. (2004). Heuristic algorithms for the multiple-choice multidimensional knapsack problem. *Journal of Operational Research Society*, 55(12):13231332.

Hifi, M., Michrafy, M., and Sbihi, A. (2006). A reactive local search-based algorithm for the multiple-choice multi-dimensional knapsack problem. *Computational Optimization and Applications*, 33:271285.

Karmacharya, A., Cruz, C., Boochs, F., and Marzani, F. (2009). Archaeokm : toward a better archaeological spatial datasets. In *Computer Applications and Quantitative Methods in Archaeology (CAA), Williamsburg, Virginia, USA*.

Karp., R. M. (1972). Reducibility among combinatorial problems. *Complexity of Computer Computations*, page 85103.

Kirkpatrick, S., Gelatt, C., and Vecchi, M. (1983). Optimization by simulated annealing. *Science*.

Lagoudakis, M. (1996). The 0-1 knapsack problem: An introductory survey. Technical report, The Center for Andvanced Computer Studies, University of Southwestern Louisiana.

Lau, H. and Lim, M. (2004). Multi-period multi-dimensional knapsack problem and its applications to available-to-promise. In *Proceedings of the International Symposium on Scheduling (ISS), Hyogo, Japan*, page 9499.

Martello, S. and Toth, P. (1990). *Knapsack problems: algorithms and computer implementations*. John Wiley & Sons, Inc., New York, NY, USA.

Picot-Clemente, R., Cruz, C., and Nicolle, C. (2010). A semantic based recommender system using a simulated annealing algorithm. In *Fourth International Conference on Advances in Semantic Processing*.