Simulation Results for a Daily Activity Chain Optimization Method based on Ant Colony Algorithm with Time Windows

Imad SABBANI[1]
Computer Sciences, Faculty of sciences and techniques
Hassan II University, Franche-Comté University
Casablanca Morocco, Besançon France

Bouattane Omar[2]
Computer Sciences, Faculty of sciences and techniques
Hassan II university
Casablanca, Morocco

Domokos Eszetergar-Kiss[3]
Transportation and vehicle engineering
Budapest university of Technology and Economics
Budapest, Hungary

***Abstract*—In this paper, a new approach is presented based on ant colony algorithm with time windows in order to optimize daily activity chains with flexible mobility solutions. This flexibility is realized by temporal and spatial change of activities achieved by travellers during one day. With the injection of flexibility concept of time and locations, the requirements for such a transport system are high. However, our method has shown promising results by decreasing 10 to 20% the total travel time of travellers based on combining and comparing different transport modes including the private transport as well as the public transport and by choosing the optimal set of activities using our method.**

***Keywords*—Component; ant colony optimization; daily activity chain; travel salesman problem; simulation**

I. Introduction

When planning daily travels, recent geospatial information systems support travellers to schedule their activities. However, these systems do not consider multiple aspects related to the preferences of the users and constraints of the activity locations (e.g. opening hours, duration needed, …) that travellers find useful or interesting. Travellers tend to combine the use of private and public transport services with the purpose to capitalize on the strengths of the various systems while avoiding their weaknesses. These combinations need to take up the challenges related to the inherent complexity of urban transportation networks as well as the range of dynamic elements [1] implicated in such systems. Furthermore, the high growth of web based applications and its user base have become source for large volume of data available online which may be helpful to generate some service suggestions in real time for users by collecting their interests, locations and preferences. Meanwhile, the growing of mobility demands and the need for cheaper and less intrusive ways to collect activity based travel diaries have defined new and innovative directions of transportation research which aim is to decrease the journey time and distance of travellers, to improve the quality and efficiency of transportation services and to optimize all aspects of transportation planning process in an automated and intelligent way [2-5].

Travel behaviour can be seen from another perspective by considering some parameters that affect greatly the trip characteristics as efficient tools of reducing travel distance, travel time and mobility needs of citizens and by the same to feed the activity-based models. We can distinguish generally three main parameters: 1) transport and land use policies [6], 2) spatial development patterns [7], and 3) socio-economic and demographic factors [8]. This can be realized by implementing intelligent activity planning methods, especially the organization of daily activity chains. For example, in [9], authors have shown the effects of several life-cycle events on the changes in time allocation in activities and associated travel. Other researchers [10] have presented the development of a mobility assistance system, which gathers information from timetables and real time information systems in public transportation. This system is connected to mobility services like car sharing, knows the users schedule and only presents relevant information for the ongoing situation. It supports the user's travel behaviour by providing information on mode, route or alternative starting times of trips. In [11], characteristics and limits of the methods used by current trip planners for path generation were presented. According to authors, experiments confirm that the use of individual, instead of average (group), utility path functions improve the path advice performance.

More recently, some authors have paid more attention to the organization of daily activity chain using the agent-based simulation in order to introduce individual decision making, flexible interaction between agents and multi-level modelling and simulation. For example, in [12], researchers have presented a simulation toolkit MATSIM to capture the patterns of people's activity scheduling and participation behaviour in order to optimize the locations of secondary activities like shopping and leisure. Travel time and costs are evaluated in this work using a fitness function and optimized by means of genetic algorithms. In [13], the authors proposed a model for an intelligent agent for adapting daily activity schedule with respect to external events, by introducing the necessity of flexible human decision making for producing realistic daily plans. Other works in the same field can be found in [14-16].

The purpose of this paper is to propose an application of the ant colony optimization meta-heuristic algorithm in order to resolve the traveling salesman problem with time windows by finding the minimum cost tour in which all point of interests are visited only once within the time windows required, involving the constraint of flexibility in time and space. Based on these, this paper is organized as follows: Section II presents a state of art of the main concepts proposed by researchers to solve the daily activity chain problem, in addition to the both concepts used in our model, traveling salesman problem and the ant colony algorithm. Section III details the proposed approach and describes the developed algorithm. Section IV shows the data used in our model and the experimental results. Concluding remarks are given in Section V.

II. Traveling Salesman Problem

The traveling salesman problem (TSP) is one of the most intensively studied problems in optimization. It's a N-P hard algorithmic problem [17] which consists on a salesman who wishes to find the shortest path between a set of points or locations that all of them must be visited with the challenge of finding the minimum total distance (i.e. cost, time, …) travelled. The salesman is supposed to visit each city only once, by starting from a certain location (e.g. hometown) and returning to the same place. The TSP can be represented by a complete weighted graph G= (V, E) with V being the set of n nodes (locations of activities) and E being the set of edges linking the nodes in the graph G. Thus, each edge E is associated with a given weight D*ij* which represents the distance between cities i and j. In symmetric TSP, it may be important to emphasize that the distances between towns/cities are the same and independent of the direction of traversing the edges, which mean that D*ij*=D*ji* for every pair of nodes forming an undirected graph. However, in the asymmetric TSP, distances may be different in both directions, due to one-way or other reasons, forming a directed graph. Hence, the TSP can be formulated as the following formulation:

We consider a graph as defined in this section, let:

V: set of nodes, $i \in V, j \in V$ and $i,j = 1,..., n$

We assume that the following data is available:

d_{ij}: distance (weight) of arc from node *i* to node *j*

We can label the activity locations with numbers 1,…,n and define:

$$X_{ij} = \begin{cases} 1 & \text{The path goes from activity location i to j} \\ 0 & \text{Otherwise} \end{cases}$$

Then we can define the TSP problem as:

$$\text{Min} \sum_{i \in V} \sum_{j \in V} d_{ij} X_{ij} \quad (1)$$

Subject to the constraints:

$$\sum_{i \in V} X_{ij} = 1 \quad (2)$$

$$\sum_{j \in V} X_{ij} = 1 \quad (3)$$

TABLE I. TSP

Method	Works
Ant colony optimization	[18,19]
Genetic algorithms	[20,21]
Neural networks	[22,23]
Memetic algorithm	[24,25]

The objective function (1) minimizes the total cost of all travels. Constraint (2) describes that only one activity location can be visited at each step of the day. Constraint (3) stipulates that every node is visited one and only one time during all the circuit.

Recently, many different approaches have been applied for solving the TSP. Table 1 shows the main methods used by researchers in order to solve the TSP.

III. Ant Colony Optimization

Ant Colony Optimization (ACO) is a population-based metaheuristic which was introduced in the early 1990s by Marco Dorigo and colleagues as a new technique for solving hard combinatorial problems [26]. The development of this algorithm was inspired by the behaviour of real ants which utilizes the pheromone communication medium, known as stigmergy, to search for the best path between the nest and a source of food. It's known as an indirect way to communicate through a chemical substance which is evaporative and accumulative. The representation of the ACO meta-heuristic in pseudo-code is as follows:

```
Procedure ACO_Metaheuristic
                Initialization
        While (not_termination)
                generateSolutions ()
                daemonActions ()
                pheromoneUpdate()
         end while
end procedure
```

At the initialization step, all d*ij* which represent the euclidean distance between an activity location I and J are initialized to a constant value *τ0*. After that, each ant presents a solution for the problem asynchronously and concurrently via the generateSolutions function by moving on the graph through adjacent intersections and by building paths. Thus, at each iteration *i* of the algorithm, each ant applies a local decision of its current state proportional to the quality of the solution represented. The probability for an ant K at an activity location I to choose to move to J is by applying the following probabilistic transition rule:

$$\begin{cases} p_{ij}^k(t) = \dfrac{(\tau_{ij}(t)^\alpha)(\eta_{ij})^\beta}{\sum_{i \epsilon V} \left(\tau_{ij}(t)\right)^\alpha (\eta_{ij})^\beta} & if\ j \epsilon J_k(i) \\ 0 & Otherwise \end{cases}$$

where η_{ij} is the heuristic visibility of edge (i, j) which equals to $1/d_{ij}$, where d_{ij} is the distance between an activity location *i* and *j*. V is a set of cities which remain to be visited when the ant is at an activity location *i*. α and β are are two adjustable positive parameters that control the relative weights of the pheromone trail and of the heuristic visibility.

The ants tend generally to choose the shorter path with a higher probability on which the pheromone trail increase faster and have a greater amount of pheromone than the longer one. However, some ants can choose the longer path with a lower probability. This concept which make the algorithm avoid a local optimum, and always search and try some different feasible solutions. At the end of each iteration, the total travelling time is reduced by minimizing the objective function:

$$f^k(t) = \sum_{i}^{S}\sum_{j}^{S} t_{ij}$$

After all ants have built their tours, and the objective function is evaluated, the pheromone is updated on all arcs as the following rule:

$$\tau_{ij}(t) \leftarrow (1-\rho).\,\tau_{ij}(t) + \rho.\,\tau_0$$

Where, τij (t) is the quantity of pheromone at time t on the arc (i, j); ρ is a parameter controlling pheromone decay such that 0 < ρ < 1; and τ0 is the initial value of pheromone on all arcs.

After all ants have finished their tour, the pheromone evaporation process starts on all arcs. Each ant k deposits a quantity of pheromone $\Delta\,\tau_{ij}^k$ (t) on each arc by the following rule:

$$\Delta\,\tau_{ij}^k\,(t) = \begin{cases} \frac{1}{L^k(t)} & if\ (i,j)\ \epsilon\ T^k(t) \\ 0 & Otherwise \end{cases}$$

Where $T^k(t)$ is the tour completed by an ant k at iteration t, and L(t) is its length. The evaporation process has the advantage of delaying and avoiding the convergence towards a locally optimal solution. This process makes the algorithm able to explore different paths during the search process.

A. Use of ACO in Solving TSP with Time Windows

The traveling salesman problem with time windows (TSPTW) is the problem of finding a minimum cost path that visits each of a set of destinations exactly once, where each activity location must be visited within a given time window, considering the duration needed to perform the activities that the traveller may find useful or interesting. The main purpose of TSPTW is to minimize the sum of travel time on the path suggested. Many constraints are required in a TSPTW problem which can be formulated as:

$$(a_i + b_i + c_i)x_{ij} < y_j \qquad \forall\text{(i,j)},$$

where $x_{ij} \in \{0, 1\}$ is a decision variable with a value of 1 if arc (*i*, *j*) is visited and 0 otherwise; $a_i = \max\{a_i, t_i\}$, with t_i indicating the time the agent arrives at node i; a_i indicates the time point at which the agent can start to serve the node i; and a_i is the service time at node *i*.

In this study, we developed our algorithm with two main objectives *g*, *h*. One is to respect the time window for all steps of the travel by avoiding to violate the deadlines. The other is to minimize the tour duration. For this purpose, we consider a new transition rule based on the Equation II represented as:

$$\begin{cases} p_{ij}^k(t) = \dfrac{(\tau_{ij}(t)^\alpha)(g_{ij})^\beta(h_{ij})^\gamma}{\sum_{i\epsilon J(i)}\left(\tau_{ij}(t)\right)^\alpha(g_{ij})^\beta(h_{ij})^\gamma} & if\ j\ \epsilon\ J_k(i) \\ 0 & Otherwise \end{cases}$$

Where $\alpha\ \beta\ \gamma$ are controlled parameters set respectively by realizing many tests to define their value. g_{ij} presents the constraint that an ant should visit the node with an arrival time closer to its upper time-window constraint, in order to avoid the lateness. However, h_{ij} represents the amount of the waiting time at a node j where the ant wants to visit. The pheromone is then updated as follows:

Procedure ACS-TSPTW

*/*Initialisation*/*
Set BestCost := ∞;
Set τij := τ0; for all (*i*,*j*)
Set all ant at the depot
Set for all (*i*,*j*) Δτij (t) = 0

*/*Iterative loop*/*
For every ant k=1 to m $\{m\ the\ number\ of\ nodes\}$

*/*Construct a Solution*/*
Compute local heuristics h_{ij}, g_{ij}
Choose the node j to move to based on the probability(I)
Delete j from the next destinations
Cost := Cost of the current solution;
If (Cost < BestCost)
BestCost := Cost;
BestSol := current solution;
EndIf
EndFor

*/*Local pheromone updating*/*
For each move (i, j) in solution BestSol
Update the trail level τij (III);
EndFor

*/*Evaluation*/*
If the stop criterion is met then stop, otherwise go to (1)

Where, BestCost is the entire travel time of solution BestSol which refer to the best tour computed by an ant k. The process is repeated by starting again with all ants until the stop criterion is met.

IV. Experimental Results and Discussion

In this section, we present the numerical results obtained by our method. First, the data used in our model is described. Then ACO-TSPTW settings and results are discussed.

A. External Database

In this study, the Budapest Maps is downloaded for an offline use in our local storage. Different information were collected (i.e. longitude, altitude, type, description, opening and closing time) from several databases (i.e. Google Maps, POI services, OSM, …) for the functioning of the system. All this data is summarized in a central database. For each task the processing time required is provided to achieve it. Table 2 shows an example of a daily activity chain used in our approach.

In addition, the Google API is used to get the directions between locations. It receives a direction request and returns the whole path. The travel time is the main parameter to be optimized, but other parameters such as distance, number of turns are also taken into account. It provides 2500 free requests per day, computed as the total of client-side and server-side queries. When using Google API, we needed to specify the transportation mode to use. The following travel modes are all supported [27]:

- **DRIVING** (Default) indicates standard driving directions using the road network.
- **BICYCLING** requests bicycling directions via bicycle paths & preferred streets.
- **TRANSIT** requests directions via public transit routes.
- **WALKING** requests walking directions via pedestrian paths & sidewalks.

B. *Design of Experiment*

Our ACO-TSPTW metaheuristic framework was implemented in Matlab and all runs were taken on a PC (3,2 GHz CPU and 1G RAM). We tested our approach up to 50 time in order to reach the best configuration possible for our settings. After many trials, the optimum combination of parameters was found is as follows: number of iteration is 100, number of ants is 25, α is 0.1, β is 2.2 , ρ is 0.85, q_0 *is* 0.99. We tried to get the fewest number of ants and iterations. These factors impact directly the solution quality and the CPU time which represent an important means of measuring the performance of the algorithm.

C. *Simulation Results*

The simulations are implemented based on two main scenarios. The first one is the basic one where only the fix schedule with fix activities in time and space is considered. However, the second one introduces the flexibility concept in time and space. For this purpose, we affect label 1,2,3 or 4 to each task as seen in Table 3, in order to define the fixed and flexible activity locations.

After running our algorithm many times, Fig. 1 reports the relativity time needed to perform a whole of a same daily activity chain. We can distinguish that flexibility in time or space can reduce the time needed to visit all activity locations by around 15% less than the fix schedule. Thus, the combined mode using an ideal version of free floating car-sharing (i.e. an available car reachable within 5 minutes walking) and public transport at the same day is always the optimum solution. However, the processing time to achieve these results reveals that the combined mode is extremely higher than the other modes as seen in Table 4.

Table 4 shows the results of our ACO-TSPTW using the different data sets in order to evaluate the robustness of our algorithm. The average of the total travel time of 5 replications is summarized with the CPU time required for each instance. In addition to, we represent a caption of our framework results in Fig. 2.

TABLE II. DAILY ACTIVITY CHAIN EXAMPLE

Point of interest	Latitude	Longitude	Opening time	Closing time	Duration
Sports Center	47.47976	19.057713	06 :00 :00	23 :00 :00	45 min
Hair-dresser	47.483183	19.053911	09 :00 :00	20 :00 :00	20 min
School	47.478556	19.056560	07 :00 :00	19 :00 :00	360 min
Mall	47.436183	19.041442	09 :30 :00	20 :30 :00	60 min
Pub	47.47914	19.08833	16 :30 :00	02 :00 :00	120 min
Home	47.433035	19.075762			

TABLE III. FLEXIBILITY LABELS

Label	Flexibility
0	None
1	Space
2	Time
3	Space and time

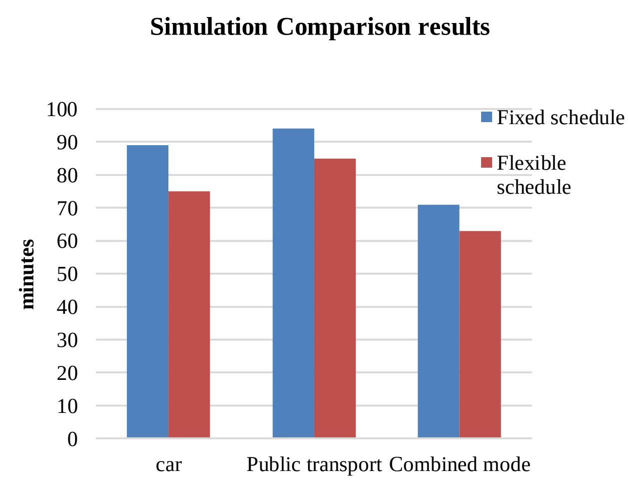

Fig 1. Comparison Simulation Results.

TABLE IV. PERFORMANCE COMPARISON OF OUR SIMULATION RESULTS

Problem instances	CAR				Public Transport				Combined			
	Fix		Flexible		Fix		Flexible		Fix		Flexible	
	CPU	Average	CPU	Average	CPU	Average	CPU	Average	CPU	Average	CPU	Average
R101	24s	93min	62s	82min	22s	112min	60s	102min	45s	75min	92s	70min
R102	21s	79min	55s	56min	18s	88min	62s	80min	42s	73min	102s	68min
R103	31s	83min	70s	72min	17s	92min	55s	81min	47s	78min	110s	65min
R104	21s	102min	68s	93min	19s	90min	67s	86min	41s	63min	104s	59min
R105	25s	89min	63s	70min	22s	88min	70s	75min	38s	65min	120s	55min
R106	26s	90min	83s	81min	20s	96min	72s	87min	41s	68min	107s	62min

Fig 2. Daily Activity Chain Example using a Car.

D. Discussion

This study focuses on the comparison of the Ant colony algorithm performances when solving the complex activity chain problem with the inclusion of flexibility in time and space. From our experiments, we realized that the flexibility concept decreases around 10% to 20% the total time needed to perform a whole of a daily activity chain in all cases. In addition to, the combined mode can be considered much faster than the others, but it requires more processing time by around 100% to 400% than the car and the public transport modes. However, these results don't depend only on the time and location of activities, but it can also depend on some other parameters (i.e. weather, peak hours, the cities size, …) that can change from a city to another one and can enormously impact the total travel time needed, although the processing time will dramatically increase.

V. CONCLUSION

The aim of this study is to present a new daily activity chain approach based on ant colony algorithm with time windows. The new concept of flexibility in time and space is introduced, which considerably decreases the total travel time by 10 to 20%. However, the CPU time needed to perform the introduction of flexibility concept has increased dramatically but remains reasonable and manageable. Regarding the obtained results, working on an online mode can be really interesting and innovative. Improvements of these first results are in progress.

REFERENCES

[1] Waidringer, Jonas. Complexity in transportation and logistics systems an integrated approach to modelling and analysis. Chalmers University of Technology, 2001.

[2] Giovanna, C., Giuseppe, M., Antonio, P., Corrado, R., Francesco, R., & Antonino, V. (2016). Transport models and intelligent transportation system to support urban evacuation planning process. IET Intelligent Transport Systems, 10(4), 279-286.

[3] CHANG, Hsien-Tsung, CHANG, Yi-Ming, et TSAI, Meng-Tze. ATIPS: automatic travel itinerary planning system for domestic areas. *Computational intelligence and neuroscience*, 2016, vol. 2016, p. 1.

[4] LI, Jing-Quan, ZHOU, Kun, ZHANG, Liping, *et al.* A multimodal trip planning system incorporating the park-and-ride mode and real-time traffic/transit information. In : *Proceedings ITS World Congress*. 2010. p. 65-76.

[5] CERNY, Ron. Method and system of planning and/or managing a travel plan. U.S. Patent No 9,009,167, 14 avr. 2015.

[6] Litman, T., & Steele, R. (2017). Land use impacts on transport. Victoria Transport Policy Institute.

[7] PUCHER, John et BUEHLER, Ralph. Why Canadians cycle more than Americans: a comparative analysis of bicycling trends and policies. Transport Policy, 2006, vol. 13, no 3, p. 265-279.

[8] Heesch, K. C., Giles-Corti, B., & Turrell, G. (2015). Cycling for transport and recreation: associations with the socio-economic, natural and built environment. *Health & place*, *36*, 152-161.

[9] SHARMEEN, Fariya, ARENTZE, Theo, et TIMMERMANS, Harry. Incorporating time dynamics in activity travel behavior model: a path analysis of changes in activity and travel time allocation in response to life-cycle events. *Transportation Research Record: Journal of the Transportation Research Board*, 2013, no 2382, p. 54-62.

[10] HILGERT, Tim, KAGERBAUER, Martin, SCHUSTER, Thomas, *et al.* Optimization of individual travel behavior through customized mobility services and their effects on travel demand and transportation systems. *Transportation Research Procedia*, 2016, vol. 19, p. 58-69.

[11] NUZZOLO, Agostino et COMI, Antonio. Individual utility-based path suggestions in transit trip planners. *IET Intelligent Transport Systems*, 2016, vol. 10, no 4, p. 219-226.

[12] HORNI, Andreas, SCOTT, Darren M., BALMER, Michael, *et al.*Location choice modeling for shopping and leisure activities with MATSim: Combining micro-simulation and time geography. *Working paper/IVT*, 2008, vol. 511.

[13] RINDSFÜSER, G., KLÜGL, F., et FREUDENSTEIN, J. Multi-agent simulation for the generation of individual activity programs. *Application of Agent Technology in Traffic and Transportation*, 2004, p. 165-180.

[14] AULD, Joshua et MOHAMMADIAN, Abolfazl Kouros. Activity planning processes in the Agent-based Dynamic Activity Planning and Travel Scheduling (ADAPTS) model. *Transportation Research Part A: Policy and Practice*, 2012, vol. 46, no 8, p. 1386-1403.

[15] SUN, Zhongwei, ARENTZE, Theo, et TIMMERMANS, Harry. A heterogeneous latent class model of activity rescheduling, route choice and information acquisition decisions under multiple uncertain events. *Transportation research part C: emerging technologies*, 2012, vol. 25, p. 46-60.

[16] Dickey, J. W. (2017). Metropolitan transportation planning. Routledge.

[17] GAREY, Michael R. et JOHNSON, David S. Computers and intractability: A guide to the theory of npcompleteness (series of books in the mathematical sciences), ed. *Computers and Intractability*, 1979, vol. 340.

[18] SHANG, Gao, LEI, Zhang, FENGTING, Zhuang, *et al.* Solving traveling salesman problem by ant colony optimization algorithm with association rule. In : *Natural Computation, 2007. ICNC 2007. Third International Conference on*. IEEE, 2007. p. 693-698.

[19] DORIGO, Marco et GAMBARDELLA, Luca Maria. Ant colonies for the travelling salesman problem. *biosystems*, 1997, vol. 43, no 2, p. 73-81.

[20] FREISLEBEN, Bernd et MERZ, Peter. A genetic local search algorithm for solving symmetric and asymmetric traveling salesman problems. In : *Evolutionary Computation, 1996., Proceedings of IEEE International Conference on*. IEEE, 1996. p. 616-621.

[21] MOON, Chiung, KIM, Jongsoo, CHOI, Gyunghyun, *et al.* An efficient genetic algorithm for the traveling salesman problem with precedence constraints. *European Journal of Operational Research*, 2002, vol. 140, no 3, p. 606-617.

[22] AVŞAR, Bihter et ALIABADI, Danial Esmaeili. Parallelized neural network system for solving Euclidean traveling salesman problem. *Applied Soft Computing*, 2015, vol. 34, p. 862-873.

[23] POTVIN, Jean-Yves. State-of-the-art survey—the traveling salesman problem: A neural network perspective. *ORSA Journal on Computing*, 1993, vol. 5, no 4, p. 328-348.

[24] Gutin, G., & Karapetyan, D. (2010). A memetic algorithm for the generalized traveling salesman problem. *Natural Computing*, *9*(1), 47-60.

[25] Mavrovouniotis, M., & Yang, S. (2011). A memetic ant colony optimization algorithm for the dynamic travelling salesman problem. *Soft Computing*, *15*(7), 1405-1425.

[26] A. Colomi, M. Dorigo, and V. Maniezzo, "Distributed optimization by ant colonies," Proceedings of the 1st European Conference on Artificial Life, pp.134-142, 1991.

[27] https://developers.google.com/maps/documentation/javascript/directions