Agent Approach Based on Reinforcement Learning for Split Delivery Vehicle Routing Problem

Farida Ben Belkacem[1] and Diaf Moussa[2]

Laboratoire de Vision Artificielle et Automatique des Systme (LVAAS)
Electrical Engineering and Computer Science Faculty
M. Mammeri university
Tizi-Ouzou, Algeria
[1]faridabenbelkacem87@gmail.com , [2]mous.diaf@gmail.com

Abstract. This paper deals with the Split Delivery Vehicle Routing Problem (SDVRP) for which several approaches still continue to be proposed. The objective is to minimize the total fleet cost for satisfying the all customers demands. For that, in this work, we introduce the Multi-Agents System after splitting the problem into sub-problems where agents cooperate, communicate and learn by introducing the reinforcement learning. Q-Learning is used to find the appropriate customers to be served by an only one vehicle. For evaluating the performances of the proposed method, we use some available SDVRP benchmark problem sets. The comparison of the obtained results with those given by the most known methods shows that the approach we propose is very promising especially in execution time and in minimizing the number of vehicles.

Keywords: SDVRP, multi-agent system, Q-Learning

1 Introduction

The vehicle routing problem (VRP) is formulated, first, in 1959 [1], and it is still current to date in transportation, logistics and supply chain management. This problem had first been formulated in different ways. The most common cases is expressed by fulfilling some simple requirements. First, vehicles having to serve the customers demands begin and end their routes at a central depot. Second, when the customers are located on the same route, the sum of their demands must not exceed the vehicle capacity. Third, each demand must be delivered by only one vehicle. However, this formulation can be adapted according to the particularities of the posed problem. In all cases, the intended solution consists in minimizing the global travel cost that is to say, the distance, the time of the way and the number of involved vehicles [1][2]. In this paper, we consider the variant known as Split Delivery Vehicle Routing Problem (SDVRP). In this case, a customer can be served by more than one vehicle when his demand exceeds the vehicle capacity. This requires the exploitation of multiple vehicles for satisfying

each customers demand while assuming that an unlimited fleet of vehicles of the same capacity Q is available. The problem is to determine whose customers are to be served by the same vehicle and what route this vehicle has to follow, while minimizing both, the operational cost of the fleet means, the travel distance and the number of involved vehicles. Note that a vehicle can serve more than one customer. Mathematically, the SDVRP is defined as a graph $G = (V, E)$ where $V = \{0, 1, ..., N\}$ is the set of the nodes and $E = \{(i, j),\ \ i, j \in V\}$, the set of the edges. The node "0" represents the depot and the other nodes represent the customers. The cost of passing an edge $(i, j) \in E$ is denoted by c_{ij} and it represents the Euclidean distance between i and j and it is assumed that c_{ij} is non-negative. A demand d_i is associated to each customer $i \in V - \{0\}$. As for the other problems of combinatory optimization, many works have tried to solve the SDVRP by using exact methods, heuristics and metaheuristics. However, the exact methods resolve only instances of limited size as it is proposed by Lee et al. [2] who formulated the problem as a dynamic programming. The problem being NP-hard, various metaheuristics [3] have been introduced. In this field Dror and Trudeau[4], Archetti and al. [5] and Boudia and al. [6] have used, respectively, the local search, tabu search algorithm and a genetic algorithm. The results obtained by the metaheuristics are known as to be good but they do not guaranty the exact optimality. Thus, A.S. Xanthopoulos and al. [7] have proposed a multi-agent based framework to vehicle routing in relief delivery system. They have considered architecture of two levels. The high level is occupied by the fleet manager which launches the auctions and the low level by vehicle agents which implement the random strategy in order to answer the bids. For including negotiation in various manners such as the contract net protocol and market-based, Multi-Agent Systems (MAS) have been introduced. In this way, Fisher and al. [8] have modeled transportation vehicles and companies by using the contract net protocol (CNP) as the task allocation protocol. M Zargayouna and al. [9] have proposed multi-agent approach which follows the CNP hybridized with a heuristic to resolve VRP with Time Windows (VRPTW). Jiri Vokrinek and al. [10] have proposed a model based on CNP composed of three types of agents: a task agent to handle the demands and invocate of allocation, an allocation agent to decide which vehicle services a given customer and the vehicle agent to plan the route and optimization. In addition, some works use the MAS approach for solving the VRP. In this field, M. Simon and al. [11] have proposed an agent-based distributed framework where the agents cooperate using a direct peer to peer asynchronous message passing protocol. These agents execute in parallel and each one implements various metaheuristics and local search combination. D. Barbucha and al. [12] have resolved the VRP by two types of agents. The first type is the set of optimization agents that act in parallel and communicate between them to improve the solution. The second type saves the intermediate results on the common memory. In another work, D. Barbucha [13] has proposed an approach based on the cooperation between a set of optimization agents which implement heuristics and local search method. The agents cooperate according to the asynchronous or synchronous mode with the learning mechanism. Some

other works including the use of MAS in transportation can be found in the general survey paper proposed by P. Davidsson and al. [14].
Note that the metaheuristics developed to the VRP are more suitable to the static case of the problem. In the VRP dynamic case, when the demands arrive while the delivery has already started, the metaheuristics become note appropriate. Otherwise, in this later case, the algorithms must restart in order to find the new solution. Note also that the already proposed agent approaches are intended to resolve VRP only when the customers' demands are less than the vehicles capacity. For our knowledge, if MAS have been already applied with success in several fields, They have not been applied for solving the SDVRP. By taking into account that the algorithms must be often reinitialized when using metaheuristics and, more importantly, when the problem is complex, dynamic and geographically distributed, the use of the MAS is an interesting option. So, for solving the SDVRP, we propose a new approach based on reinforcement learning. We implement a Q-Learning algorithm [15] which is based on the Q-value function and converges towards the optimal value. It exploits the temporal structure of the problems (current state, action, following state, reward) contrary to some other methods as the metaheuristics that search only optimal solutions.
In the following of this paper, the second section includes the description of the proposed MAS approach and the role of each agent considering the coordination, the communication and the learning of the agents. In the third section, we present the algorithm details of the proposed method. The tests and results are given in the fourth section. Finally, the conclusion and perspectives are given is the fifth section.

2 Multi-Agent system in proposed method

The architecture of the proposed MAS applied to the SDVRP is illustrated in Fig.1. It includes a coordinator agent, a planning agent, a learning responsible agent and learning agents. The two-way arrows indicate that the messages are exchanged between agents in both directions. The customers' geographical coordinates as well as those of the central depot are presented in a matrix called position matrix M_p. The customers and their demands are presented in a flow matrix M_f.
The coordinator agent is the user interface. It loads the data contained in M_p and M_f, splits the customers' demands and sends them to agents, to the planning agent and to the learning responsible agent. Then, this coordinator agent receives and groups the results of these agents and generates a global plan.
The planning agent receives data from the coordinator agent and generates the partial planning for deciding on the number of necessary vehicles to serve a given customer as well as the rounds number for each vehicle.
The learning responsible agent is the agent which undertakes the learning. It receives data from the coordinator agent and launches the learning agents one after another so that it does the learning to each vehicle. Whenever an agent ends the learning, it sends the results to the learning responsible agent. This lat-

ter updates its data and launches another agent until all customers are served. Then, it sends the result (partial plan) to the coordinator agent. The learning agents are launched by the learning responsible agent and they learn on the basis of the Q-learning method. Their number, not known in advance, is dependent on the customer's number and their demands.
For each vehicle, each learning agent defines the set of customers to insert in its route by taking into account the minimization of the travel distance. At the end of the learning, it sends the results (partial plan) to the learning responsible agent.

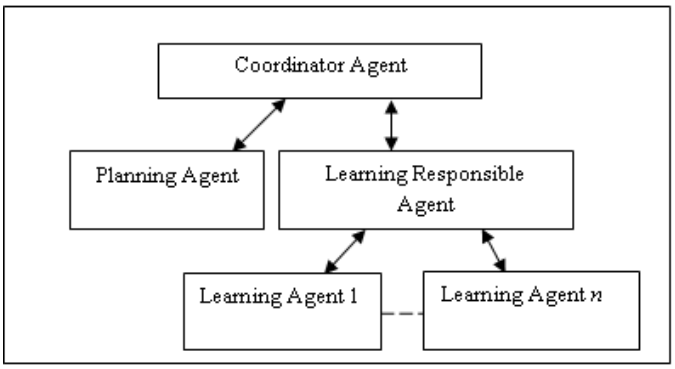

Fig. 1. Architecture of proposed MAS

3 The Proposed Algorithm

Once the system agents and their roles are defined, we split the problem into two sub-problems by subdividing the flow matrix M_f into two matrixes M_1 and M_2 expressed as following:

$$M_f = M_1 + M_2 \tag{1}$$

where $M_1$is the matrix on which quantity MODULO (capacity of the vehicle) is equal to zero and M_2 is a matrix on which the quantity is lower than the capacity of the vehicle. M_2 is obtained by M_f MODULO (capacity of the vehicle). As example, let us suppose that we have 6 customers A, B, C, D, E, F with their corresponding demands as it shown in table 1. We also suppose the vehicle capacity is 100. Table 2 and 3 show the calculated matrixes M_1 and M_2 respectively. From table 2, the planning agent fixes the necessary vehicles number for serving each customer. From table 3, the responsible learning agent groupes the customers to be served by the same vehicle and defines the necessary number of vehicles. M_1 is used by the planning agent and M_2 is used by the learning responsible agent.
The agents used for solving the SDVRP are the planning agents and the learning

Table 1. Example of flow matrix M_f

A	**B**	**C**	**D**	**E**	**F**
100	60	280	489	450	1246

Table 2. Example of flow matrix M_1

A	**B**	**C**	**D**	**E**	**F**
100	0	200	400	400	1200

Table 3. Example of flow matrix M_2

A	**B**	**C**	**D**	**E**	**F**
0	60	80	89	50	46

responsible agent. The planning agent receives the inputs M_1 and M_p and set the necessary vehicles number that must serve each customer as well as it set the rounds number for each vehicle. The learning responsible agent receives the inputs M_2 and M_p. Since the quantity of the fleet which must be transmitted between a depot and a customer is lower than the capacity of the vehicle, the responsible learning agent launches learning agents to find appropriate customers to be inserted in a route of a given vehicle while the quantity is lower or equal to the capacity of the vehicle. As soon as the learning agent ends the learning, it sends the results to the learning responsible agent which updates M_2 by removing the chosen customers. Then another learning agent is launched, and so on, until all the customers' requests are satisfied.
To do that, we have applied the reinforcement learning by using the Q-learning algorithm. We remind that the method of Q-Learning [15] is based on the function of Q-value and converges towards the optimal value. This algorithm works by interaction with the environment in the following way (Fig.2). At each step, in the current state s, the agent chooses an action a according to policy $\prod$. It executes the action a, receives the reward r and observes the following state s'. Then, it updates the values of $Q(s, a)$ according to Equation (2):

$$Q(s,a) \leftarrow (1-\alpha)Q(s,a) + \alpha[r + \gamma \max_{a' \in A} Q(s^{'}, a^{'})] \tag{2}$$

In this algorithm, α is the learning rate and γ, the discount factor. As it is shown in expression (2), the update consists in removing a part of the previous $Qvalue(s, a)$ and in adding a part of the new evaluation with the immediate reward and the value of the next state.

3.1 Application Of Q-Learning

Before all, let us define the states, the actions and the function of the reward for the learning strategy based on Q-Learning for the SDVRP. These actions are:

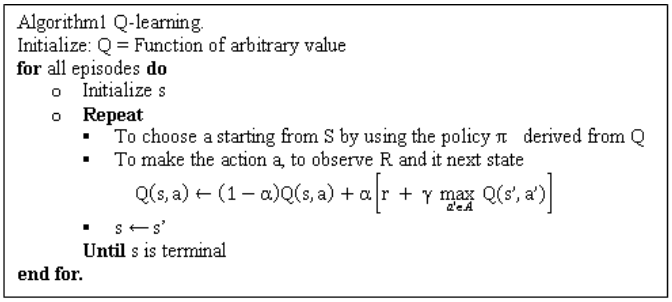

Fig. 2. Algorithm of Q-Learning

- I, the current node of a vehicle.
- J, the node of arrival of a vehicle (central depot).
- A, the measure taken at a given time.

Let (i, j) be a set of states between current nodes i and arrival nodes j , and $A(i, j)$, the set of agent actions candidates for the next node. Note that these actions are customers who are candidates to be served by the same vehicle. The reward function $R[(i, j), a]$ is the report of the distance covered by the vehicle going from node i towards node a by having, for destination, the central depot j. $R[(i, j), a]$ is obtained by using the following expression:
$R[(i, j), a] = distance[i][a]/(vehicle\ capacity - demand\ of\ customer\ a)$. The $Q - value$ function updates according to the equation (2). Initially we take:

$$QValues[(i, j), a] = R[(i, j), a]/(1 - discount\ factor).$$

where i is the current state, a is the action, and j the central depot.
For the choice of the actions, we have used the strategy (policy) $\xi - greedy$. Most of the time, the action with the highest estimated reward, called the greediest action, is chosen. From time to time, an action is selected with a probability ξ. This strategy ensures that, if enough trials are done, each action will be tried several times. Thus, the ensuring optimal actions are discovered. The learning agents learn by following the algorithm of Q-learning applied to the SDVRP (Fig.3).

4 Computational Experiment

The proposed method in which various agents learn to solve the SDVRP is carried out by using the platform MadKit where agents are coded in Java 1.6 language. The characteristics of the used computer are $2GB$ RAM and Intel (R) Core (TM) 2 Duo processor, 2.66 GHZ, 2.67 GHZ.
In order to validate the developed MAS and to evaluate the system performances, computational experiments have been carried out. The approach has been tested on several sets of SDVRP benchmark. Table 5 reports the experimental results.

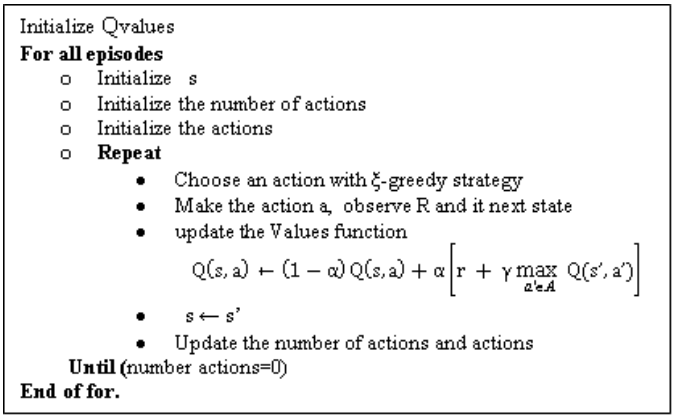

Fig. 3. Algorithm of Q-learning used by the learning agents

Its structure is as follow: the first column corresponds to the tested instance name, the second column, is subdivided into the vehicles number, the covered distance and the overall execution time, refers to the best known results which are extracted from references [17] [18] and the last column shows the results obtained with the method we proposed. Before running the system, some parameters related to the Q-learning algorithm have been fixed empirically (Table 4).

Table 4. The Q-learning Parameters

Rate training	**discount Factor**	**Epsilon**	**Number of episodes**
0.05	0.9	0.01	2000

The number of episodes is set at to 2000. This parameter influences on the obtained results especially on the convergence of the learning towards the optimal solution. Epsilon (ξ) is set at to 0.01. It is used for the selection of actions. The discount factor γ is related to the Q-Learning algorithm. It is better to set at at large value (i.e. near to 1.0). In our case, γ=0.9. Concerning the rate training α representing the capacity of the agent instant memorization, a large value can be fixed as well, but this may conduct the agent to give much importance to the first chosen actions which could constitute a bad choice in terms of optimal behavior. In our case, $\alpha = 0.05$ reduces the time of learning and guarantees optimal results.

As it can be seen in table 5, the obtained results are very interesting compared to the best ones published in literature. Let us note that, in the most instances, the travel distances obtained by our method are very near the best ones published in literature, but they are not the best ones. This is due to the weakness of Q-learning in front of the optimization methods. Concerning the employed

vehicles number, the results given by the method we propose are competitive by comparison with the best known results published in the literature. For some instances, the results are comparable. This is mainly due to the learning agents which find, for each vehicle, the set of customers to insert on its route by maximizing the filling of vehicles. The execution time is also reasonable for almost the tested instances. This is among the advantages of the SMA where several entities are carried out at the same time.

5 Conclusion

In this paper, we have presented an approach based on the multi-agents system for the resolution of transport problem which is the split delivery vehicle routing problem. Unlike the classic approaches, the use of SMA has allowed us to subdivide the problem into sub-problems by cooperating and communicating several entities and especially to develop a learning strategy of these entities by using Q-Learning with dynamic actions. The obtained results are very interesting and competitive comparatively to those given in literature concerning especially the vehicles number and the CPU time. However, the approach we propose requires improvements like taking into account the case limited fleet and unpredictable situations as vehicles breakdown during the service considered for solving the same problem when the data are dynamic.

References

1. Dantzig, G., Ramser, J. : The truck dispatching problem, Manag.Sci. 6(1), 80-91 (1959)
2. Lee,C.G., Epelman,M.A., White III,C.C, Bozer, Y.A. : A Shortest path approach to the multiple-vehicle routing problem with split pick-ups, Transportation Research B, 40: 265-264 (2006)
3. Archetti, C., Speranza, M.G.: Vehicle routing problem with split deliveries, International Transactions in Operational Research. 19(2012) 3-22 (2012)
4. Dror, M., Trudeau, P.: Savings by split delivery routing, Transportation Science 23, 141-145 (1989)
5. C. Archetti, A. Hertz and M.G. Speranza, A Tabu Search algorithm for the split delivery vehicle routing problem, Transportation Science, 40, 64-73 (2006)
6. Boudia, M., Prins, C., Reghioui, M.: Misc: An Effective Memetic Algorithm with Population Management for the Split Delivery Vehicle Routing Problem. In: Bartz-Beielstein, T., Blesa, M.J., Blum, C., Naujoks, B., Rlli, A., Rudolph, G. , Samuels, M.. (eds) Hybrid Metaheuristics, Lecture Notes in Computer Science 4771. Springer, Berlin, pp. 16-30, (2007)
7. Xanthopoulos, A. S., Koulouriotis,D. E.: A Multi-agent based framework for vehicle routing in relief delivery systems, Operations Research/Computer Science Interfaces Series (54), Humanitarian and Relief Logistics Research Issues, Case Studies and Future Trends (2013)
8. Fischer, K., Mller, J.P., Pischel, M.: Cooperative transportation scheduling: An Application domain for DAI, Journal of Applied Artificial Intelligence, Special Issue on Intelligent Agents, 10 (1996)

9. Zargayouna, M., Balbo, F., Seemama, G.: A multi-agent approach for the dynamic VRPTW, Esaw (2008).
10. Vokrinek, j.: Agent Towards Vehicle Routing Problem, International Foundation for Autonomous Agents and Multiagent Systems (2010)
11. Simon, M., Djamila, O., Patrick , B., Ender, O., Angel, A. J, Edmund, K.B: a multi-agent based cooperative approach to scheduling and routing, European Journal of Operational Research (2016),
12. Barbucha, D., Jedrzejowicz, P.: An agent-based approach to vehicle routing problem, International Journal of Applied Mathematics and Computer Science, 4(2), 538-543 (2007)
13. Barbucha, D.: Search modes for the cooperative multi-agent system solving the vehicle routing problem, Neurocomputing (88), 13-23 (2012).
14. Davidsson, Paul., Henesey, L., Ramstedt, L., Trnquist, J., Wernstedt, F. : An analysis of agent-based approaches to transport logistics, Transportation Research Part C, (13),255-271 (2005)
15. Dayan, P., Watkins, C.: Q-learning, Machine Learning, pp. 8(3): 279-292 (1992)
16. Watkins, C.: Learning from delayed rewards,Doctoral thesis, Cambridge University,Cambridge, England (1989)
17. Ping, C., Bruce, G., Xingyin, W., Edward, W.: A novel approach to solve the split delivery vehicle routing problem", Intl. Trans. in Op. Res. 00 1-15 (2016)
18. Leonardo, B., Sergio, G., Francisco J.N.: A Randomized Granular Tabu Search heuristic for the split delivery vehicle routing problem, Annals of Operations Research, Volume 222, Issue 1, pp 153-173 (2014)

Table 5. Results of Multi-Agent System for SDVRP

	Best Known results			Proposed approach		
problem	Number of vehicles	Travel distance	CPU Time(sec)	Number of vehicles	Travel distance	CPU Time(sec)
eil22	4	375.28	3	4	466.00	1.2
eil23	3	568.56	2	3	820.52	0.9
eil30	3	497.53	8	3	684.63	2.6
eil33	4	826.41	8	4	1060.84	1.1
eil51	-	524.61	55	5	836.23	1.7
eilA76	10	823.89	29	10	1184.85	2.8
eilB76	14	1009.04	20	14	1600.00	5.4
eilC76	8	738.67	33	8	1142.07	3.0
eilD76	7	684.53	122	7	1271.41	3.1
eilA101	8	812.51	295	8	1855.63	6.9
eilB101	14	1076.26	173	13	2979.96	7.2
S51D1	-	459.50	6	3	707.45	2.0
S51D2	-	709.29	22	9	1568.21	2.3
S51D3	15	948.06	31	14	1764.58	1.7
S51D4	-	1562.01	54	26	2730.21	1.9
S51D5	-	1333.67	61	23	2395.72	2.3
S51D6	41	2169.10	54	37	3166.76	2.0
S76D1	-	598.94	252	4	969.13	2.6
S76D2	-	1087.40	59	15	2070.59	3.2
S76D3	-	1427.86	136	22	2496.78	3.7
S76D4	37	2079.76	111	35	3114.32	4.6
S101D1	-	726.59	2126	5	1089.34	6.0
S101D2	-	1378.43	219	20	3852.63	8.4
S101D3	31	1874.81	168	30	4477.34	10.1
S101D5	-	2791.22	238	47	5401.05	14.9
$P01\_110$	-	459.50	256	3	829.88	2.203
$P01\_1030$	11	757.15	256	10	1631.03	1.688
$P01\_1050$	16	1005.75	35	15	1963.61	2.171
$P01\_1090$	-	1488.85	48	25	2586.23	1703
$P01\_3070$	26	1481.71	45	24	2642.16	1.937
$P01\_7090$	41	2156.14	55	38	3401.75	2.109
$P02\_110$	-	617.85	12	4	1049.28	3.813
$P02\_1030$	-	1109.62	42	16	1883.52	6.125
$P02\_1050$	24	1502.05	61	23	2339.55	4.391
$P02\_1090$	-	2298.58	104	37	3422.66	4.766
$P02\_3070$	39	2219.97	89	37	3390.01	4.719
$P02\_7090$	62	3223.40	872	55	4529.16	6.250
$P03\_110$	-	752.62	409	6	2445.74	6.421
$P03\_1030$	-	1458.46	180	21	3834.38	9.657
$P03\_1050$	33	1996.76	248	32	4526.45	10.831
$P03\_1090$	-	3085.69	391	52	5606.58	14.703
$P03\_3070$	53	2989.30	364	49	5613.07	18.453
$P03\_7090$	82	4387.32	62	74	6558.63	18.391